Using Sentiment to Detect Bots on Twitter: Are Humans more Opinionated than Bots?

John P. Dickerson[†,‡]
[†]Carnegie Mellon University
Pittsburgh, PA, USA
Email: dickerson@cs.cmu.edu

Vadim Kagan[‡]
[‡]Sentimetrix, Inc.
Bethesda, MD, USA
Email: kagan@sentimetrix.com

V.S. Subrahmanian[⋆]
[⋆]University of Maryland
College Park, MD, USA
Email: vs@cs.cmu.edu

***Abstract*—In many Twitter applications, developers collect only a limited sample of tweets and a local portion of the Twitter network. Given such Twitter applications with limited data, how can we classify Twitter users as either bots or humans? We develop a collection of network-, linguistic-, and application-oriented variables that could be used as possible features, and identify specific features that distinguish well between humans and bots. In particular, by analyzing a large dataset relating to the 2014 Indian election, we show that a number of sentiment-related factors are key to the identification of bots, significantly increasing the Area under the ROC Curve (AUROC). The same method may be used for other applications as well.**

I. Introduction

The openness of Twitter's platform allows for, and even promotes, programs that automatically post. These "bots" post content ranging from helpful (e.g., recent news stories or public service announcements) to malicious (e.g., spam or phishing links). Such malicious bots on Twitter have become a nuisance, even recently triggering a long diatribe in the New Yorker [1]. They are alleged to help political candidates skew public perception; for instance, botornet.net asserts that former US Presidential candidate Newt Gingrich gained over a million followers on Twitter through the use of bots, a charge Mr. Gingrich is reported to have denied. Similarly, Hill [2] reports that "Facebook thinks 83 million of its users are fake" and that "up to 29.9% of Barack Obama's 17.82 million [Twitter] followers and 21.9% of Mitt Romney's 814,000 followers may be fake". In short, there is now a widespread belief that bots constitute a significant part of the social media world—and that many of them are malicious in their intent.

In this paper, we study the problem of identifying bots on Twitter *from an application perspective*. Few real-world applications analyze all of Twitter. Rather, most applications focus on those portions of the Twitter tweet database and the Twitter network that are relevant to them. For example, if a politician were interested in tracking the recent Indian elections, he would probably identify a set of topics of interests (TOI) consisting of names of relevant politicians and relevant political parties. The identification of bots may then be based on a relevant subset of tweets and a relevant subset of the Twitter network that are *TOI focused*. Compared to past work [3] on bot identification in Twitter, TOI-focused applications may not have a huge network with which to work, especially as reconstructing a network from Twitter's open API can pose some challenges. In such industrial applications, there is a critical need to identify bots:

1) For instance, in an election application we built, we needed to predict the expected number of supporters for a particular candidate. But in this prediction, we needed to discount for bots.
2) We worked with a company that wanted to identify the most influential Twitter users on a suite of topics of interest to the company. They wanted to be sure that the measure of influence discounted for bots (and of course that bots were not included in the answer).

In contrast to past work, we approach this task from a new viewpoint, namely analyzing the novel semantic feature of *tweet sentiment* in human and bot accounts. To our knowledge, no prior work has used tweet sentiment to separate human and non-human users on Twitter. We present SentiBot, a sentiment-aware architecture for identifying bots on Twitter. We test it on a real dataset that does *not* include malicious accounts already suspended by Twitter's fielded bot detection algorithms, but—as validated by our human labelers—still includes bots that slipped through their filter. Including sentiment-aware features in the classification process improves accuracy on these "harder" classification cases, where presently fielded algorithms fail.

A. Previous Work

There has been recent interest in the detection of malicious and/or fake users from both the online social networks and computer networking communities. For instance, Wang [4] looks at graph-based features to identify bots on Twitter, while Yang, Harkreader, and Gu [5] combine similar graph-based features with syntactic metrics to build their classifiers. Thomas et al. [6] use a similar set of features to provide a retrospective analysis of a large set of recently-suspended Twitter accounts. Boshmaf et al. [7] instead *create* bots (rather than detecting them), claiming that 80% of bots are undetectable and that Facebook's Immune system [8] was unable to detect their bots. Lee, Caverlee, and Webb [9] create "honeypot" accounts to lure both humans and spammers into the open, then provide a statistical analysis of the malicious accounts they identified. In computer networks research, the detection of Sybil accounts in computer networks has been applied to social network data; these techniques tend to rely on the "fast mixing" property of a network—which may not exist in social networks [10]—and do not scale to the size of present-day social networks (e.g., SybilInfer [3] runs in time $O(|V|^2 \log |V|)$, which is intractable for networks with millions users).

Most relevant is recent work by (Twitter employee and anti-spam engineer) Chu and colleagues [11], [12], which uses graph-theoretic, syntactic, and some semantic features to classify humans, bots, and cyborgs (human-assisted bots) in a Twitter dataset. With the exception of a few projects like Chu et al. [12], bot detection research has focused only on graph-theoretic properties of social networks and syntactic—not semantic—content of tweets. Thus, this work primarily focuses on the semantic feature of *tweet sentiment* (at the individual user or neighborhood level). While work exists that uses tweet sentiment to gauge public opinion (see, e.g., [13]–[16]), to our knowledge, this is the first time sentiment has been used for account verification and classification.

B. Our Contribution

In this paper, we propose SentiBot, an architecture and associated set of algorithms that automatically identifies bots on Twitter by using—for the first time—a combination of features including *tweet sentiment*. At a high-level, SentiBot can draw features from:

1) Sophisticated *sentiment analysis techniques* [13], [14], [17], [18] to analyze sentiment on Twitter on a per-user basis on a variety of topics;
2) Neighborhood-aware semantic metrics on a per-user and per-topic basis (e.g., "does this user tend to disagree with the users she follows?");
3) Syntactic tweet metrics associated with a user such as the average number of hashtags, repeated tweets, and a variety of other such proven statistics;
4) Other semantic linguistic models such as the online version of latent Dirichlet allocation (LDA) [19] to identify and consider the topics discussed by various Twitter users; and
5) Graph-theoretic and a variety of other metrics.

This defines a set of *contextual variables* associated with each user in a dataset $\mathcal{D}$; SentiBot then automatically computes values of each contextual variable for each user in the dataset. We will discuss these variables and the SentiBot architecture in detail in Sections II and III. Then, given a labeled training dataset, SentiBot builds an ensemble of classifiers using standard machine learning techniques, and optimizes for maximum precision or recall. In Section IV, we conduct a detailed experimental analysis of the SentiBot bot detection framework applied to a real-world dataset of 7.7 million Indian political tweets by over 555,000 users and show that it can achieve acceptable and realistic precision and recall numbers, finding bot accounts that were not caught by Twitter's fielded malicious account detection algorithms.

Our key result shows that of the 25 top contextual variables in determining whether a user is a bot or not, 19 are sentiment-related. Moreover, in our real world India Election Dataset (IEDS), 14 of these 19 sentiment related variables are tied to a specific topic of interest to the application. This suggests two things: (i) sentiment plays a significant role in identifying bots, and (ii) taking the topics of interest to an application into account is very important for identifying bots associated with a specific application.

II. Dataset and Architecture

The architecture of our SentiBot system is shown in Figure 1. We will now overview each portion of the architecture.

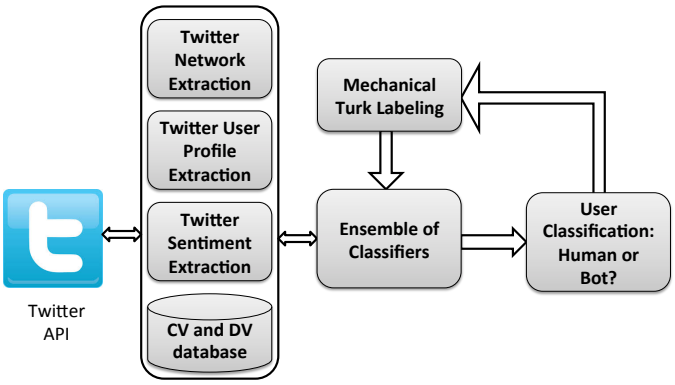


Fig. 1: Architecture of the SentiBot System

We use the "India Election Dataset" (IEDS), a real-world dataset that was collected from July 15, 2013 to March 24, 2014. It consists of information on over 550,000 Twitter accounts and 7.7 million tweets on the topic of the upcoming 2014 Indian election. A tweet is considered "on topic" (and included in IEDS) if it includes one of a number of keywords pertaining to the election (e.g., Indian political parties like "Shiv Sena" or "BJP", politicians like "Rajnath Singh" or "Nitish Kumar", etc.) or if it was tweeted by a user who frequently tweets about such political keywords. We discuss the set of these "topics of interest" (TOI) later in this section. Critically, there exists a strong incentive to deploy bots in the IEDS dataset, as noted by the Indian press' coverage of the use of bots and fake accounts during this election cycle [20].

Twitter Sentiment Extraction. SentiBot used the Twitter API to first identify a set $\mathcal{U}_0$ of users during the selected time frame who had tweeted on at least one "topic of interest" in a set TOI. A TOI is any set of terms. In the case of IEDS, the TOI consisted of politicians and political parties involved in the election. For each day d, each user u, and each $t \in$ TOI, we calculated the sentiment score $\mathsf{SS}(d, u, t)$ of user u on topic t, averaged across all tweets on topic t posted by the user on that day. We used SentiMetrix's commercially available sentiment scoring engine [17], [18] that assigns a value between -1 ("maximally negative") and $+1$ ("maximally positive") to score the intensity of sentiment on topic t in a tweet. Of course, this scoring engine can be swapped out with other similar scoring engines such as those due to Barbosa and Feng [13] or Agarwal et al. [14].

Network Database. SentiBot then examines the profiles of users in $\mathcal{U}_0$. For each user in $\mathcal{U}_0$, it identifies the set $\mathcal{U}_1 = \{u' \mid (\exists u \in \mathcal{U}_0)$ such that either u follows u' or u' follows $u\}$ of followers and followees of individuals in $\mathcal{U}_0$. A set $\mathcal{U}_2 = \{u'' \mid (\exists u' \in \mathcal{U}_1)$ such that either u' follows u'' or u'' follows $u'\}$ is also constructed. The complete set of users considered is $\mathcal{U} = \mathcal{U}_0 \cup \mathcal{U}_1 \cup \mathcal{U}_2$ and the network constructed is the subgraph of the Twitter follower network induced by the set $\mathcal{U}$ of users. There is an edge from user u to u' in this subgraph if u follows u'. In the case of IEDS, there were over 40 million edges considered in total.

User Profile Extraction. For each user in $\mathcal{U}$, we extract a profile by using not only their Twitter profile, but also by looking at their position in the network, their tweets, and various aspects of their behavior. This will be described in great detail in Section III. Note that throughout this paper, we use the term "Twitter profile" to refer to the user profile displayed on Twitter and the term "User Profile" to refer to the profile we define in this paper.

CV and DV database. Using the three preceding components, SentiBot builds a database whose rows correspond to users and whose columns correspond to a set of contextual variables (CVs). These variables describe various aspects of the user's sentiments, topics he may tweet about, properties of the user with respect to the local or global IEDS network, and more. All of these aspects of a user's behavior are stored in a database. In addition, *for our training set alone*, the rows of users corresponding to our training data include one dependent variable (DV) BOT set to 1 if the user is a bot and 0 otherwise. *The identification of the contextual variables—specifically which ones are relevant in deciding whether an entity is a bot or not in datasets such as ours—is one of the main contributions of this paper.*

Ensemble of Classifiers. We used well-known classifiers to answer the question: Is a user u a bot or not? In total, we tried six high-level classifiers including support vector machines (SVM) for classification, Gaussian naive Bayes, AdaBoost, gradient boosting, random forests, and extremely randomized trees; we discuss the optimization and (hyper)parameter selection process in detail in Section IV. The result of applying these classifiers is a labeling and confidence bound, for each user in a test dataset, of whether that user is a bot or not.

Training and Testing Datasets. We randomly selected a training set of 897 users from the India Election Dataset (IEDS). The true number was larger, but some of the users had been flagged as malicious accounts by Twitter prior to our human annotation, so these users were discarded from our dataset. We note that this makes our bot classification problem more difficult, as Twitter is also incentivized to remove malicious accounts; our dataset only includes those bots that slipped through Twitter's filter. We used Amazon Mechanical Turk to get five sets of annotations on whether these users were bots or not. We provided each annotator with information on each of the 897 users and asked them to review the users' Twitter profiles and tweets to decide whether each user was likely a bot or human. We also asked each Mechanical Turk user to write at least three sentences of natural text describing why a user is labeled as a bot or human; this provided a hedge against the risk that the Mechanical Turkers would not do a thorough job. As yet another step to ensure they did a good job, we included a monetary incentive for reporting correctly, adding a bonus of USD $1 for every correctly identified bot (from a subset of previously determined bot accounts) and a negative bonus of USD -$1 for each incorrectly identified bot account (from a subset of previously determined bot accounts). In the event of disagreement across the Mechanical Turk results, we took a majority vote across the set of 897 users to determine a final label. We call this labeled training data the *annotated IEDS* dataset.

As discussed in more detail in Section IV, we performed a 50/50 train/test split on the annotated IEDS. We learned classifiers on the training split (via 5-fold cross-validation and a grid search over appropriate hyperparameters for each classifier) and tested on the independent test set. We report results in Section IV.

III. SentiBot User Profile

In this section, we define the profile of a user that is used by SentiBot. The measures are divided into four categories, each described in a subsection below.

1) *Tweet Syntax:* This class of variables captures various aspects of the construction of tweets. *This class of variables has been used before (see, e.g., [5]) and is not claimed as a novel contribution.*
2) *Tweet Semantics:* This class of variables captures various aspects of the content of tweets from a semantic and sentiment orientation and how they have changed over time. *This class of contextual variables, as well as (3) and (4) below, consists of several sentiment-related variables that have never been used before.*
3) *User Behavior:* This class of variables captures various aspects of the user (derived from his Twitter profile when available) as well as several sentiment-related variables showing the user's sentiments and their variance over time.
4) *User Neighborhood:* This class of variables captures various aspects of the immediate neighborhood around the user, together with what those neighbors are discussing and what sentiments they hold on various topics. It includes several sentiment-related contextual variables that capture measures of the user's sentiments (on various topics) as compared to those of his neighbors.

In short, a large number of variables in our study are sentiment related; out of the 145 variables in our study, 135 were sentiment-related and 128 were TOI-related. Note that TOI-specific variables may have a sentiment component and hence these two counts can have overlap. For instance, a contextual variable such as "Average Sentiment (BJP)" captures the average sentiment (in IEDS) on the topic "BJP" (referring to Bharatiya Janata Party, an Indian political party).

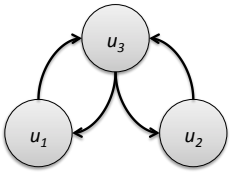
A. Tweet Syntax

We used the following variables to describe the syntax of tweets posted by a user. These are common to many other studies (see, e.g., Yang, Harkreader, and Gu [5]).

1) **Average number of hashtags:** This is the average number of hashtags used in a tweet posted by user u.
2) **Average number of user mentions:** This is the average number of other users mentioned by user u in his tweets.
3) **Average number of links:** This is the average number of links present in tweets by user u.
4) **Average number of special characters:** This is the average number of special characters (e.g., emoticons) present in tweets by user u.

TABLE I: Toy dataset consisting of three users tweeting about a single topic t_1 (left) and their local network (right).

User	Tweet ID	Topic t_1
u_1	1	$+0.2$
u_2	2	$+0.8$
u_1	3	$+0.3$
u_2	4	$+0.2$
u_3	5	-0.1
u_1	6	-0.2
u_1	7	$+0.3$



B. Tweet Semantics

We used the following variables to describe the semantics of tweets posted by a user u. To our knowledge, each of the sentiment-based features presented is novel with respect to determining whether or not a Twitter account is legitimate. Before describing each feature, Table I presents a toy dataset consisting of just three users, u_1, u_2, and u_3, in a very small network, tweeting about a single topic t_1. We will refer to this example dataset when describing some of the sentiment-based features below.

1) **Average topic sentiment(t):** We identified the average sentiment expressed by user u on each topic $t \in \mathsf{TOI}$ during the entire life-time of the IEDS dataset. This average sentiment varied on a -1 to $+1$ scale. Using the example dataset in Table I, user u_1's average topic sentiment about topic t_1 is $\frac{1}{4}(0.2+0.3-0.2+0.3) = +0.15$, or slightly positive.
2) **Average topic sentiment($overall$):** This is the average sentiment expressed by the user, averaged over all topics about which the user expressed any sentiment.
3) **Sentiment Polarity Fractions(t):** For each topic $t \in \mathsf{TOI}$, we tracked the percentage of tweets posted by user u that were positive (resp. negative) on topic t. Again referring to Table I, user u_1's positive sentiment polarity for topic t_1 is $\frac{3}{4}$, with negative sentiment polarity fraction $\frac{1}{4}$. We note that the positive and negative sentiment polarity fractions need not add up to 1 due to tweets with neutral sentiment on or no mention of topic t_1.
4) **Sentiment Polarity Fractions($overall$):** This represents the average sentiment polarity fraction for the user, aggregated across all topics about which the user expressed sentiment.
5) **Contradiction Rank(t):** For each topic $t \in \mathsf{TOI}$, let x_t^+ be the fraction of tweets showing *positive* sentiment toward t out of the set of u's tweets showing *any* sentiment toward t. Similarly, let x_t^- be the fraction of user u's tweets showing *negative* sentiment toward t out of the set of u's tweets showing *any* sentiment toward t. Similarly, define y_t^+ and y_t^- to be the fraction of tweets across all users showing positive or negative sentiment toward topic t out of all tweets showing any sentiment toward topic t. Then define the *contradiction rank* of user u to be $\mathsf{CR}(u,t) = x_t^+ y_t^- + x_t^- y_t^+$. Intuitively, a high contradiction rank means that the majority of users tend to disagree with u on topic t. Referring to the example dataset in Table I, we calculate $\mathsf{CR}(u_1, t_1)$ as follows. The user tended to post positively about t_1, so $x_{t_1}^+ = \frac{3}{4}$ and $x_{t_1}^- = \frac{1}{4}$. Similarly, the entire dataset also tended to post positively about t_1, so $y_{t_1}^+ = \frac{5}{7}$ and $y_{t_1}^- = \frac{2}{7}$. Thus, $\mathsf{CR}(u_1, t_1) = \frac{3}{4}\cdot\frac{2}{7} + \frac{1}{4}\cdot\frac{5}{7} \approx 0.393$, a fairly low contradiction rank.
6) **Agreement Rank(t):** For each topic $t \in \mathsf{TOI}$, compute x_t^+, x_t^-, y_t^+, and y_t^- as above. Then define the *agreement rank* of user u to be $\mathsf{AR}(u,t) = x_t^+ y_t^+ + x_t^- y_t^-$. Intuitively, a high agreement rank means that the majority of users tend to agree with u on topic t. Thus, from the example in Table I, $\mathsf{AR}(u_1, t_1) = \frac{3}{4}\cdot\frac{5}{7} + \frac{1}{4}\cdot\frac{2}{7} \approx 0.607$, which is notably higher than $\mathsf{CR}(u_1, t_1)$, as expected.
7) **Dissonance Rank:** Given a user u and agreement and contradiction ranks for each topic t, we define a new aggregate measure, the dissonance rank of u, as

$$\mathsf{DR}(u) = \sum_{t \in \mathsf{TOI}} \mathsf{CR}(u,t)/\mathsf{AR}(u,t).$$

8) **Positive Sentiment Strength(t):** For each user u and topic t, given the set of all tweets by u that contain *positive* sentiment about t, this is the average such positive sentiment strength about that topic. Consider the toy example shown in Table I. User u_1 tweeted positively about topic t_1 three times, in the first, third, and seventh tweets. In these three tweets, the user's sentiment scores on topic t_1 are $+0.2$, $+0.3$, and $+0.3$, and so the positive sentiment score for this user u_1 is approximately 0.267.
9) **Negative Sentiment Strength(t):** For each user u and topic t, given the set of all tweets by u that contain *negative* sentiment about t, this is the average such negative sentiment strength about that topic. Again, consider the toy example shown in Table I. User u_1 tweeted negatively about topic t_1 once, at sentiment value -0.2; hence, the average negative sentiment strength of user u_1 about topic t_1 is -0.2. *In general, positive sentiment strength and negative sentiment strength* can be viewed as measures of the strengths of opinions that a user expresses. Some users do not write neutral tweets; they may only write tweets where strong opinions are expressed. Others may mostly write tweets when they only have strong negative opinions, and so forth. These two variables try to capture such characteristics of users.
10) **Positive Sentiment Strength($overall$):** For each user u, this is the average positive sentiment strength over each topic $t \in \mathsf{TOI}$.
11) **Negative Sentiment Strength($overall$):** For each user u, this is the average negative sentiment strength over each topic $t \in \mathsf{TOI}$.
12) **LDA topics:** We identified a probability distribution over the space of topics, specifying the probability that the user discusses a topic. Here, the word "topic" is not a topic of interest (which is application specific), but the set of topics identified in the entire IEDS using latent Dirichlet allocation (LDA) [19] to classify words (minus English stopwords and words that did not occur more than once) in the tweets in the IEDS into clusters. For example, we wondered if humans are more focused in terms of the topics they discussed than bots—and this variable was used

to capture some of these aspects. We did not include sentiment-based analysis for these LDA topics.

C. User Behavior

This category covers temporal aspects of the user's tweeting habits. Of the properties shown below, (5)–(9) are novel.

1) **Tweet Spread:** Given a histogram over the 24 hours of the day describing when the user tweeted, this feature calculates the entropy of that user's tweeting times. For more information on using entropy as a measure for tweet time "spread", see the motivation in Chu et al. [12].
2) **Tweet Frequency:** The average number of tweets put out by user u on a daily basis. The motivation for this variable was that perhaps users who tweet a huge amount are actually bots.
3) **Tweet Repeats:** The average number of tweets that are repeated by a user u.
4) **Geoenabled:** Are the user u's tweets associated with a geotag?
5) **Tweet Sentiment Flip-Flops:** This variable measured the amount of flip-flopping in sentiment on a given topic $t \in \mathsf{TOI}$. A sentiment "flip" occurs when user u put out a positive or negative tweet on a topic t at one time and then reversed her sentiment on the topic later. This variable measures the number of such inversions (positive to negative and vice-versa) and normalizes these inversions by the total number of tweets authored by the user. In the example in Table I, user u_1 flip-flopped twice on topic t_1 out of a total of four tweets, for a score of $\frac{1}{4}(2) = 0.5$.
6) **Tweet Sentiment Variance(t):** For each topic $t \in \mathsf{TOI}$, we measured the variance in sentiment expressed by the user over the entire IEDS dataset.
7) **Monthly Tweet Sentiment Variance(t):** As a user may have legitimately changed her opinion on a topic t during the eight months of our study, we also tracked the average variance in sentiment exhibited by user u on topic t on a monthly basis.
8) **Average topic volume histogram(t):** This variable tracked the average (monthly) volume of tweets by user u on each topic $t \in \mathsf{TOI}$ in the IEDS.
9) **Average topic volume histogram($overall$):** Averages the preceding variable across all topics $t \in \mathsf{TOI}$.

D. Network-centric User Properties

We also looked at network-centric user properties. Of the properties show below, (4), (5) and (6) are novel.

1) **In-degree:** The number of users following u.
2) **Out-degree:** The number of users that u is following.
3) **In/Out Ratio:** The number of followers of a user u divided by the number of users u is following.
4) **Neighborhood Contradiction Rank(t):** This is a version of the contradiction rank where, for each topic $t \in \mathsf{TOI}$, y_t^+ and y_t^- are computed only by looking at tweets about t posted by neighbors of a specific user u. Intuitively, a high *neighborhood contradiction rank* $\mathsf{NCR}(u,t)$ for a user u and topic $t \in \mathsf{TOI}$ signals that the neighbors of a user u tend to disagree with that user about topic t. For example, the global contradiction rank $\mathsf{CR}(u,t)$ could be high because user u maintains a (valid) point of view not reflected by the general population studied. However, one would expect her neighborhood contradiction rank $\mathsf{NCR}(u,t)$ to be lower in value because accounts in her immediate neighborhood are "birds of a feather," sharing similar sentiments.
   In the Table I example, user u_1 neighbors only u_3; thus, $y_{t_1}^+ = \frac{3}{5}$, $y_{t_1}^- = \frac{2}{5}$, and $\mathsf{NCR}(u_1,t_1) = \frac{3}{4}\cdot\frac{2}{5} + \frac{1}{4}\cdot\frac{3}{5} \approx 0.45$. This is higher than the global contradiction rank $\mathsf{CR}(u_1,t_1)$ because u_1's only neighbor (and thus entire neighborhood) tweeted negatively about topic t_1, unlike u_1 (and also the global sentiment toward t_1, which was generally positive).
5) **Neighborhood Agreement Rank(t):** This is a version of the agreement rank where, like in the neighborhood contradiction rank, y^+ and y^- are computed only by looking at tweets in a one-hop neighborhood around a user u. The *neighborhood agreement rank* $\mathsf{NAR}(u,t)$ for a user u and topic $t \in \mathsf{TOI}$ is similarly motivated. In the example from Table I, $\mathsf{NAR}(u_1,t_1) = 0.55$, which is lower than the global $\mathsf{AR}(u_1,t_1)$, as expected.
6) **Neighborhood Dissonance Rank:** This is the local version of the global dissonance rank $\mathsf{DR}(u)$. Given a user u, define the aggregate measure

$$\mathsf{NDR}(u) = \sum_{t \in \mathsf{TOI}} \mathsf{NCR}(u,t)/\mathsf{NAR}(u,t).$$

IV. Experimental Results

In this section, we describe the experimental framework used to train and test SentiBot on a real-world dataset (IEDS). We compare the best classifier that does not include sentiment-aware features to the best classifier that does include sentiment-aware features, and show on IEDS that (i) including sentiment in feature selection improves classification and (ii) some of the most important features for separating human from bot accounts are sentiment based.

A. Learning the classifiers

We use the (annotated) IEDS dataset that was described in Section II. Feature extraction was performed as in Section III, noting that features involving the global dataset were extracted from the full IEDS. We then tried the following families of classifiers: Gaussian naive Bayes, support vector machines (SVMs) for classification, random forests [21], extremely randomized trees [22], AdaBoost [23], and gradient boosting [24]. Our classifiers were built and trained on top of `scikit-learn` [25], a machine learning toolkit supported by INRIA and Google.

After data normalization, the classification problems benefited from kernel PCA [26] as an additional preprocessing and dimensionality reduction step. Kernel PCA, a generalized form of principal component analysis (PCA) performed in a reproducing Hilbert space, allows us to move away from the simple linear features of traditional PCA. The main gain seen from kernel PCA applied to our dataset was in de-noising; since (i) even humans have difficulty discerning some bot

accounts from true human accounts and (ii) semantic feature extraction is an imperfect process, our training data's feature values and class labels had considerable noise.

We performed a grid search over a wide range of hyperparameters for each of the classifiers; depending on the classifier type, we searched over reasonable settings of the number of estimators, type of estimator, learning rates, minimum number of samples per split and per leaf, maximum depth of a tree, kernel type, and kernel type parameters like γ and degree. For each classifier and setting of hyperparameters, the grid search used five-fold cross validation on the training data to fit the model and get precision and recall scores. We optimized for overall recall; thus, the classifier and hyperparameter vector with maximum recall was used as the final classifier (and then run on the independent test data for analysis).

B. Classification precision and recall

We performed the complete grid search over preprocessing techniques, classifiers and hyperparameter vectors (described above) twice: once on the full feature set presented in Section III, and once on a reduced feature set consisting of only those features that did not involve sentiment analysis. While both the with- and without-sentiment feature set problems benefited from kernel PCA as a preprocessing step, AdaBoost performed best on the reduced feature set, and gradient boosting performed best on the full feature set. Figure 2 gives receiver operating characteristic (ROC) curves for both classifiers applied to the test dataset.

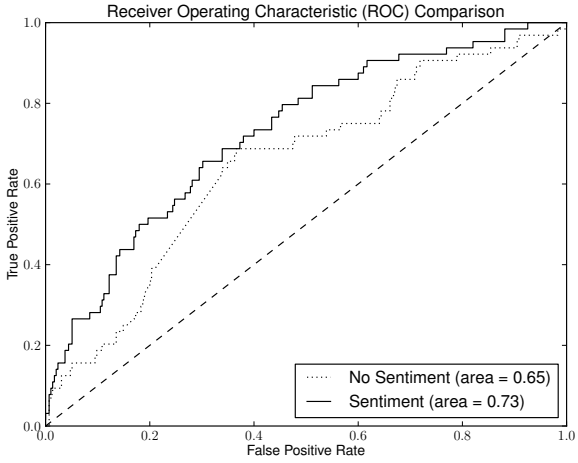


Fig. 2: ROC curves comparing the best classifiers with and without sentiment features. Area under the ROC curves are 0.73, 95% CI [0.67, 0.78] and 0.65, 95% CI [0.58, 0.71] for with and without sentiment features, respectively, and one-sided $p \ll 0.01$ for both classifiers being different than chance.

Intuitively, ROC curves visualize the performance of a classifier (in terms of true positives and false positives) as its discrimination threshold τ is tightened and loosened. As Figure 2 clearly shows, the best classifier with access to the full feature set (including sentiment variables) outperformed the best classifier with access to only the reduced feature set. The area under the curve (AUROC)—the probability that the classifier will rank a randomly chosen bot as more "bot-like" than a randomly chosen human—supports this notion, with a 0.73 for the full feature set compared to a lower 0.65 for the reduced feature set.

In our application, false positives have a significant cost; improperly accusing a legitimate account of being a bot either results in increased human verification time on the side of the social network owner or customer dissatisfaction on the side of the (suspended) account holder. This cost is especially high on our dataset, where our human labelers had difficulty discerning bots from other humans—so verifying the proper label of an account could require *multiple* human fact checkers. Figure 2 shows that, for "strict" discrimination thresholds ($\tau < 0.2$) that enforce low false positive rates, the inclusion of sentiment-based features nearly doubles the true positive rate over classifying using non-sentiment-based features. Of course, for all discrimination thresholds τ, the sentiment feature-aware classifier performed at least as well and typically much better than the classifier without sentiment features.

C. Feature importances

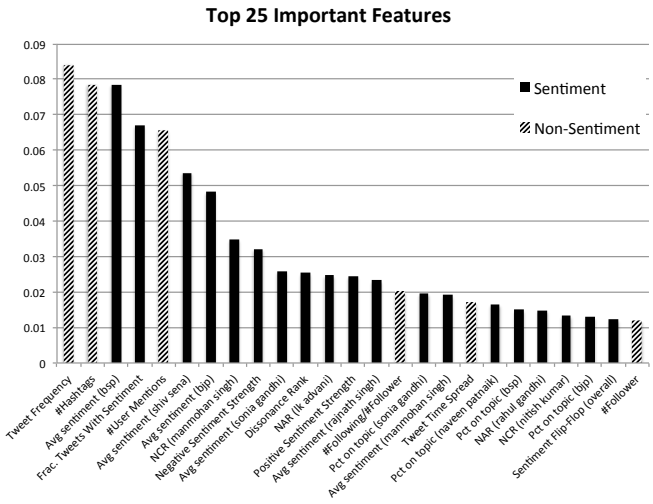
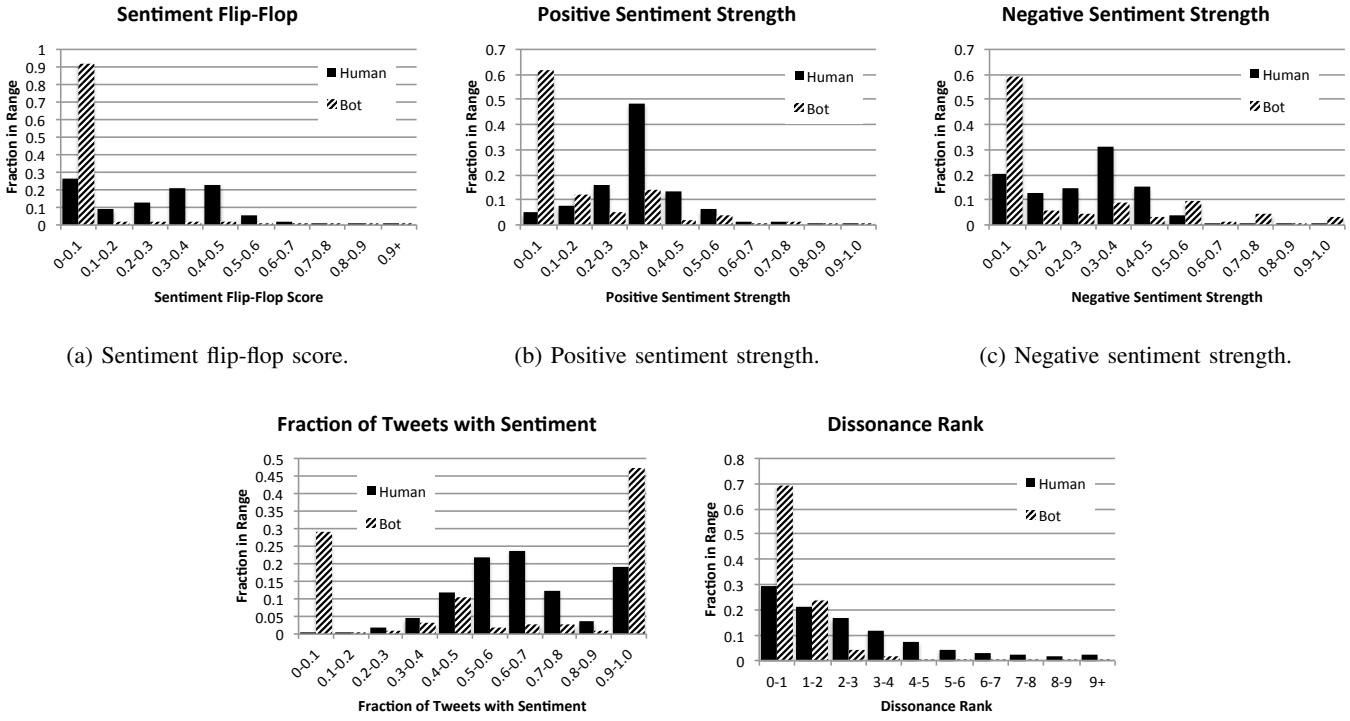


Fig. 3: Top 25 most important features in the best classifier. Sentiment-based features are shown in black, while standard features are striped.

Figure 3 shows the 25 features identified by SentiBot as being most significant in distinguishing between bots and non-bots. We see immediately that 19 of the 25 top features are sentiment related. Moreover, of these, 14 are topic specific. The topic-independent sentiment features that are significant are shown in Figure 4. Each chart in this figure shows the value of the feature on the x-axis. These values are placed into buckets based on the range of the variable (e.g., $\{[0, 0.1), [0.1, 0.2), \ldots, [0.9, 1.0]\}$ for a variable with values in the unit interval). For each bucket, we show on the y-axis the percentage of human users (in black) with that feature value in that bucket, and likewise for bots (striped). These percentages are created after classifying the full IEDS dataset and are computed from the 226,434 accounts for which our classifier was at least 90% certain of either a "bot" or "human" label. We discuss each of the topic-independent features below.



(a) Sentiment flip-flop score.

(b) Positive sentiment strength.

(c) Negative sentiment strength.

(d) Negative sentiment strength.

(e) Dissonance rank.

Fig. 4: Histograms comparing bots against humans on five significant topic-independent sentiment-aware features.

1) **Sentiment flip-flop score.** We see from Figure 4a that 92.5% of bots have a sentiment flip-flop score in the 0–0.2 range; in contrast, only 26.5% of humans have a score in this region. Almost no bots have sentiment flip-flop scores greater than 0.1. This suggests that bots rarely flip-flop on sentiment. In contrast, humans tend to flip-flop much more.
2) **Positive sentiment strength.** Figure 4b shows that 61.4% of bots have a positive sentiment strength in the 0–0.2 range. In contrast, only 4.7% of humans have scores in this range. For users with positive sentiment strength in the 0.2–0.3 range, it is hard to distinguish between bots and humans; however, most users with positive sentiment scores between 0.3 and 0.7 are humans. This suggests that when humans express positive opinions on Twitter, they tend to express strong sentiments as compared to bots.
3) **Negative sentiment strength.** Figure 4c shows that 59.2% of bots have a positive sentiment strength in the 0–0.2 range. In contrast, only 7.5% of humans have scores in this range. For users with negative sentiment strength over 0.7, it is hard to distinguish between bots and humans. However, (proportionally) most users with negative sentiment scores between 0.3 and 0.7 are humans. This suggests that when humans express positive opinions on Twitter, they tend to express strong sentiments as compared to bots.
4) **Fraction of tweets with sentiment (FTS).** Figure 4d shows that 29% of bots have an FTS in the 0–0.2 range. When the FTS is between 0.2 and 0.5, it is very hard to distinguish between humans and bots. However, when a user's fraction of tweets with sentiment is between 0.5 and 0.9, he is much more likely to be a human than a bot. 73.6% of humans have FTS scores in this range, while only 0.185% of bots have such an FTS score. And finally, few humans (19%) score between 0.9 and 1, while 57.3% of bots have FTS scores in this range. Thus, the frequency of tweets with sentiment is a highly nuanced parameter in distinguishing between bots and non-bots.
5) **Dissonance rank.** Figure 4e shows that 68.9% of bots have a dissonance rank in the 0–2 range; in contrast, only 29.1% of humans have dissonance scores within this range. However, 57.4% of humans have scores in the range 2–6, while only 30.4% of bots have scores in this region. In short, humans tend to disagree more with the entire Twitter population than bots.

V. Conclusion

In many real-world applications, developers are only able to collect tweets from the Twitter API that directly address a set of topics of interest (TOI) relevant to the application. Moreover, in such applications, developers also typically only collect a local portion of the Twitter network. As a consequence, many traditional primarily network-based methods for detecting bots [3]–[5], [11], [12] are less or not effective (e.g., if the topics are quite specific, not discussed by very popular people, or not retweeted much), since a sparse subset of the

global network and tweet database based on a set TOI is insufficient.

The SentiBot framework presented in this paper addresses the classification of users as human versus bot in such applications. In order to achieve this, SentiBot relies on four classes of variables (or features) related to tweet syntax, tweet semantics, user behavior, and network-centric user properties. In particular, we introduce a large set of sentiment variables, including combinations of sentiment and network variables—to our knowledge, this is the first time such sentiment-based features have been used in bot detection. In addition, we introduce variables related to topics of interest. We apply a suite of classical machine learning algorithms to identify (i) users who are bots and (ii) TOI-independent features that are particularly important in distinguishing between bots and humans. Based on an analysis of over 7.7 million tweets and 550,000 users associated with the recently concluded 2014 Indian election (where there were reports of social media campaigns [20]), we were able to show that the use of sentiment variables significantly improved the accuracy of our classification. In particular, the Area under the ROC Curve (AUROC) increased from 0.65 to 0.73. As an AUROC of 0.5 represents random guessing, this reflects a $\frac{(0.73-0.65)}{0.15} \approx 53\%$ improvement in accuracy. In addition, we discovered that (in our dataset):

1) Bots flip-flop much less frequently than humans in terms of sentiment;
2) When humans express positive sentiment, they tend to express stronger positive sentiment than bots;
3) A similar (but slightly more nuanced) trend holds in terms of expression of negative sentiments by humans; and
4) Humans disagree more with the general sentiment of the application's Twitter population than bots.

Our results can feed into many applications. For instance, when assessing which Twitter users are influential on a given topic, we must discount for bots—which requires methods like those presented in this paper to identify bots. When identifying the expected spread of a sentiment through Twitter, we again must discount for bots. The paper presents a general framework within which applications can identify bots using the relatively limited local data they have.

Acknowledgments. Parts of this work were funded by ARO contract W911NF-12-C-0026. Dickerson and Subrahmanian performed work as consultants for Sentimetrix, who own the resulting intellectual property.

REFERENCES

[1] R. Dubbin, "The rise of Twitter bots," *The New Yorker*, Nov 15 2013. [Online]. Available: http://www.newyorker.com/online/blogs/elements/2013/11/the-rise-of-twitter-bots.html

[2] K. Hill, "The invasion of the Twitter bots," *Forbes*, Aug 9 2012. [Online]. Available: http://www.forbes.com/sites/kashmirhill/2012/08/09/the-invasion-of-the-twitter-bots/

[3] G. Danezis and P. Mittal, "SybilInfer: Detecting sybil nodes using social networks," in *Network and Distributed System Security Symposium (NDSS)*, 2009.

[4] A. H. Wang, "Detecting spam bots in online social networking sites: A machine learning approach," in *Conference on Data and Applications Security and Privacy*. ACM, 2010, pp. 335–342.

[5] C. Yang, R. C. Harkreader, and G. Gu, "Die free or live hard? Empirical evaluation and new design for fighting evolving Twitter spammers," in *Recent Advances in Intrusion Detection*. Springer, 2011, pp. 318–337.

[6] K. Thomas, C. Grier, D. Song, and V. Paxson, "Suspended accounts in retrospect: An analysis of Twitter spam," in *Internet Measurement Conference (IMC)*. ACM, 2011, pp. 243–258.

[7] Y. Boshmaf, I. Muslukhov, K. Beznosov, and M. Ripeanu, "The socialbot network: When bots socialize for fame and money," in *Annual Computer Security Applications Conference (ACSAC)*. ACM, 2011, pp. 93–102.

[8] T. Stein, E. Chen, and K. Mangla, "Facebook immune system," in *Workshop on Social Network Systems (SNS)*. ACM, 2011.

[9] K. Lee, J. Caverlee, and S. Webb, "Uncovering social spammers: Social honeypots + machine learning," in *Annual ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 2010, pp. 435–442.

[10] A. Mohaisen, A. Yun, and Y. Kim, "Measuring the mixing time of social graphs," in *Internet Measurement Conference (IMC)*. ACM, 2010, pp. 383–389.

[11] Z. Chu, S. Gianvecchio, H. Wang, and S. Jajodia, "Who is tweeting on Twitter: Human, bot, or cyborg?" in *Annual Computer Security Applications Conference (ACSAC)*. ACM, 2010, pp. 21–30.

[12] ——, "Detecting automation of Twitter accounts: Are you a human, bot, or cyborg?" *IEEE Transactions on Dependable and Secure Computing*, vol. 9, no. 6, pp. 811–824, 2012.

[13] L. Barbosa and J. Feng, "Robust sentiment detection on Twitter from biased and noisy data," in *International Conference on Computational Linguistics (COLING)*, 2010, pp. 36–44.

[14] A. Agarwal, B. Xie, I. Vovsha, O. Rambow, and R. Passonneau, "Sentiment analysis of Twitter data," in *Workshop on Languages in Social Media*, 2011, pp. 30–38.

[15] A. Boutet, H. Kim, and E. Yoneki, "What's in Twitter: I know what parties are popular and who you are supporting now!" in *International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, 2012, pp. 132–139.

[16] S. Bouktif and M. Adel Awad, "Ant colony based approach to predict stock market movement from mood collected on Twitter," in *International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, 2013, pp. 837–845.

[17] F. Benamara, C. Cesarano, A. Picariello, D. R. Recupero, and V. S. Subrahmanian, "Sentiment analysis: Adjectives and adverbs are better than adjectives alone." in *International Conference on Weblogs and Social Media (ICWSM)*, 2007.

[18] V. S. Subrahmanian and D. Reforgiato, "AVA: Adjective-verb-adverb combinations for sentiment analysis," *IEEE Intelligent Systems*, vol. 23, no. 4, pp. 43–50, 2008.

[19] M. D. Hoffman, D. M. Blei, and F. R. Bach, "Online learning for latent Dirichlet allocation." in *Neural Information Processing Systems (NIPS)*, vol. 2, no. 3, 2010, p. 5.

[20] A. Kharpal, "Hackers target Indian election tweets with malicious spam," *CNBC*, May 10 2014. [Online]. Available: http://www.cnbc.com/id/101679812

[21] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[22] P. Geurts, D. Ernst, and L. Wehenkel, "Extremely randomized trees," *Machine Learning*, vol. 63, no. 1, pp. 3–42, 2006.

[23] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[24] J. H. Friedman, "Greedy function approximation: a gradient boosting machine," *Annals of Statistics*, pp. 1189–1232, 2001.

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

[26] B. Scholkopf, A. Smola, and K.-R. Müller, "Kernel principal component analysis," in *Advances in Kernel Methods*, 1999.