TITLE

AUTHOR

DATE

Introduction

The ubiquity of social media makes it easy for people to communicate with each other and stay informed about global events. Many platforms rely on crowdsourced information for promoting a business' services and/or products. However, this provides a vulnerability that commercial entities can easily exploit for their own interests. For example, a company could use a social media bot to promote its own products and write glowing reviews on websites such as Yelp and Google Reviews while posting negative reviews on its competitors' products. This is a lose-lose situation for both businesses and consumers; businesses must constantly defend themselves from the onslaught of negative feedback while consumers will lose trust in information from social media. Therefore, in this survey, we will discuss methods that can be used to differentiate between real users and non-human users.

At the general level, this is a problem of *anomaly detection*—finding users that are unlike the rest. Historically, there have been many approaches to anomaly detection, such as Poisson, SFP, Pearson's χ^2, and CNPP. We will analyze the pros and cons of these existing methods. Next, we will introduce state-of-the-art models that attempt to address some of the shortcomings of these more mature models and highlight their relative improvements. These current methods fall under a few major categories, including temporal, Bayesian, and group.

Background

NEEDS WORK FOR SURE

Each type of anomaly detection method introduces its own set of definitions and fundamental ideas. Temporal-based approaches primarily use the **inter-arrival times** (IAT) between postings to identify patterns that could be used for detecting anomalies. The IAT distributions between humans and bots are compared to classify users.

To fully understand group-based approaches, a **group** must be defined. A group is a mixture model of a user behavior mixture model. The **role mixture rate** is an inference of group membership and role identity of each individual inside of a group. A **group anomaly** occurs when the role mixture rate is significantly different than a normal group role mixture rate. All individuals inside of a group will be classified as suspicious if there is a group anomaly.

Models, Approaches, Algorithms

Methods developed before 2014 will be introduced here as both a basis of improvement and reference for state-of-the-art methods.

Cascading Non-homogeneous Poisson Process (CNPP) – 2009

CNPP is a variation of the double chain hidden Markov model and is used for understanding the behavior of individuals and detecting outliers. Specifically, this model is used to detect variability among individuals by using data from email communication. First, an inference algorithm is used to estimate model parameters. This model is then applied to the email data. From this experiment, the authors found that individuals can be classified into specific types and that users can be clustered together based on their model parameters which leads to easier ways to detect outliers.

Power-Law – 2005

The power-law distribution is a non-Poisson distribution that models human behavior, which consisted of short bursts of activity separated by long periods of inactivity. The explanation for this type of distribution is the fact that humans execute tasks by their own set of priorities. Thus, tasks wait an uneven amount of time before execution.

Self-Feeding Process (SFP) – 2013

Approach

The SFP model combines the benefits of the non-homogeneous Poisson Process (PP) from CNPP and the power-law distribution. This is because the non-homogeneous PP works well to model user communication where short-term communication is PP, but there are long periods of inactivity where the PP rate should change. The power-law distribution captures a user's long periods of inactivity which are seen as a heavy tail in the IAT probability density function.

Model

It attempts to accomplish four goals:

1. Realism of marginals
2. Realism of the local Poisson distribution
3. Avoiding the independent and identical distribution (i.i.d.) fallacy
4. Requirement of only a few parameters

To achieve the first goal and to better view the dataset without losing information, SFP computes the odds ratio (OR) as such:

$$OR(t) = \frac{CDF(t)}{1 - CDF(t)}$$

where the set of IATs are used to compute the OR for each percentile of the data.

For the second goal, the Poisson distribution is built into the SFP model. The third goal is obtained by building a Markov process where each even of activity influences the next event.

The Poisson process model is defined as the following:

$$\Delta_i \sim exp(1/\lambda) \ for \ i = 1 \dots n \ events$$

SFP's piecewise Poisson model updates it as:

$$\Delta_1 \sim \mu$$

$$\Delta_i \sim exp(\Delta_{i-1} + \mu/e)$$

where e is the Euler constant.

As seen from the model, the last goal is achieved, as only the mean of the data set is required as input. The model can be generalized for other applications that only need two parameters, μ and the OR slope ρ:

$$\delta_1 \sim \mu$$

$$\delta_1 \sim exp(\delta_{i-1} + \mu^\rho / e)$$

$$\Delta_k \sim \delta_i^{1/\rho}$$

Pearson's χ^2 – 2011

Approach

This paper creates a model to determine whether a Twitter account uses automation to publish tweets for spamming purposes. The authors come to the result that 16% of active accounts show automated behavior. A bot is defined as a computer program that automatically publishes a significant portion of their tweets. They believe that automated accounts have a specific timing pattern that differs from real users. For example, real users might tend to post uniformly throughout whereas automated accounts might post at the beginning or specific interval of the hour. Automated accounts tend to have non-uniform timing or excessive uniformity.

Model

The model used to detect whether the posted times match with those of real user is Pearson's χ^2 test. The test returns a p-value; if this value is below a certain threshold, the account belongs to a real user and if this value is higher than this threshold, then this account is most likely automated. For this test, the threshold was chosen to be 0.1% specifically to avoid false positives and detect accounts that post with excessive uniformity. A false positive occurs when the test says an account is automated when it is actually organic. However, keeping the threshold at 0.1% has its drawbacks as it will be difficult to classify hybrid accounts which contain both manual and automated postings. A false negative tends to be higher because of these hybrid accounts being classified as organic accounts instead of automated accounts. Another drawback of this test is that an automated account can pass this test if they post uniformly throughout the hour and second.

All accounts can display some form of automation depending on temporal factors such as time of day. In order to accurately determine automated accounts, one will need to examine tweets spanning over a couple of weeks or months, but the authors have left this to do in the future. A previous study had a model where accounts that published more that 150 tweets per day were automated. After Twitter started using this model, the number of automated accounts dropped down to 14%. When looking at only Verified accounts, only 6.9% were determined to be automated, typically due to tweeting news or reminders throughout the day. For those in Trending Topics, only 4.7% of the accounts were automated, which could be attributed to the fact that Twitter prevents automated tweets from appearing in trending topics. In the future, the authors wanted to investigate the frequency of words and compare them between automated accounts and organic accounts.

In conclusion, Pearson's χ^2 test can detect automated accounts which have distinct timing patterns, which found that 16% of public accounts are automated. The authors suggest to use this model to prevent spam and other use.

Current State-of-the-art

While state-of-the-art methods use the same basic ideas and principles of older models and fall under the same set of categories, they attempt to improve on them in some way.

Rest-Sleep-and-Comment (RSC) – 2015

Approach

This paper investigates the differences between humans and bots based exclusively on the timing of posts from two specific social media services: Reddit and Twitter. IATs are used for measuring temporal data. Using the distribution of IATs, four patterns were identified:

1. Positive correlation between consecutive IATs
2. Heavily-tailed distribution
3. Periodic spikes
4. Bimodal distribution

The patterns were common among various social media services and were identified through analyzing IAT distributions from timestamp data. A model was then designed to generate synthetic timestamps whose IAT fits the data distribution and matches all patterns of actual human users. Non-human users could then be detected using this model.

Model

RSC is a generative model that matches these patterns to classify anomalies based only on temporal activities.

First, a sequence of synthetic IAT must be generated. This is done through the Self-Correlated Process (SCorr), which is a stochastic process that generates consecutive IATs that are correlated, which is different than sequences generated from a Poisson Process or a priority queue based model. This is meant to match the positive correlation pattern from actual human data. In SCorr, the duration δ_i between two events is sampled from an exponential distribution with rate λ depending on the previous IAT δ_{i-1}. It uses the correlation parameter ρ to control the dependency between consecutive IATs. When $\rho \to 0$, SCorr reduces to a Poisson Process with rate λ and when $\rho = 1$, it reduces to a Self-Feeding Process.

Next, for modeling the bimodal IAT distribution and period spikes from actual data, three states were defined for the RSC algorithm: active, rest, and sleep. While RSC is in the active state, it generates posting events with a probability p_{post} or null events with probability $1 - p_{post}$ at every time interval δ_i^A. The intervals δ_i^A and δ_{i+1}^A are correlated and generated using SCorr. While RSC is in the rest state, null events are generated at every time interval δ^R, which contributes for incrementing the IATs between postings. While RSC is in the sleep state, a single null event is generated after a time interval δ^S, which corresponds to the time required to advance the clock time t_{clock} until the next wakeup time t_{wake}.

Because RSC is based on a 24-hour cycle, t_{clock} spans between $0:00h$ and $23:59h$. Under the assumption that $t_{wake} = 0$, the parameters t_{wake} and t_{sleep} can be replaced by a single parameter f_{sleep}, which corresponds to the fraction of the day that RSC considers as sleep-time. Using this parameter, $t_{sleep} = f_{sleep} * 24h$ when $t_{wake} = 0$. When a user is active, there is a probability p_r that it will transition into the rest state and $1 - p_r$ probability for it to remain active. While in the rest state, there is a p_a probability to transition into the active state and $1 - p_a$ probability to remain in the rest state. However, a user will always transition from rest to

sleep when $t_{wake} > t_{clock}$ and $t_{sleep} < t_{clock}$, meaning that the current clock time falls within the sleep time. After the sleep state ends, the user will always transition into the rest state.

Algorithm

To estimate the parameters for RSC, an algorithm based on fitting the histogram of the observed data IAT is used. First, synthetic timestamps are generated using the RSC model. Next, a log-binned histogram of IAT is computed for real and synthetic data. The width w_i of the i-th bin is wider than the previous bin by a fixed factor k, meaning that $w_i = w_{i-1} * k$. The counts of IAT in each i-th bin are denoted as c_i and $\hat{c}_i$ for real and synthetic data respectively. With the Levenberg-Marquardt algorithm, the parameter values θ that minimize the square difference between the synthetic and real data bin counts are calculated: $min_\theta \sum_i (c_i - \hat{c}_i(\theta))^2$. Using logarithm binning when approximating the PDF allows the parameter estimation method to match the heavy tail pattern and daily spikes found in real data.

Next, RSC-Spotter uses RSC to detect bots. It compares the distribution of IAT of each user to the aggregated distribution of IAT of all users in the dataset. Users that have an IAT distribution significantly different from the aggregate are flagged as anomalies and potential bots. It first estimates RSC parameters using the aggregate IAT data and then generates n_i timestamps, where n_i is exactly the number of timestamps for user U_i. Then, RSC-Spotter calculates the dissimilarity between the synthetic timestamp distribution with the users' actual timestamps.

Finally, it has to be determined whether or not the dissimilarity value D_i classifies that user U_i is a human or a bot. Given a training set of users labeled either as bots (positive) or humans (negative), a Naive Bayes classifier is trained to estimate the posterior probability p_{bot} that a user is a bot, using the dissimilarity values D_i as features for the classifier. If p_{bot} is greater than a decision threshold p_{thresh}, then the user is classified as a bot. The threshold p_{thresh} is estimated by assigning a cost c_{FN} to false negative errors and a cost c_{FP} to false positive errors. A false positive is when a human is classified as a bot and a false negative is when a bot is classified as human.

In conclusion, RSC was quite successful in differentiating between humans and bots. With only temporal activity, RSC was able to detect non-human users with 96.5% precision on Reddit and 94.7% precision on Twitter.

BIRDNEST – 2015

Approach

Because IATs do not completely account for bot behavior, the authors proposed BIRDNEST as a Bayesian-based method to approach the anomaly detection problem. BIRDNEST expands on the pure temporal-based approach by using both IAT and rating distributions.

When determining the suspiciousness of a user, one may first ask what the rating distribution or IAT distribution is for a given user. For example, a user that rates all ones or all fives would have a single peaked rating distribution and a user that rates all fives for its own company's products and all ones for its competitors' products would have a bimodal rating distribution. BIRDNEST approaches user rating-fraud detection by solving these questions with Bayesian inference. The user?s rating distribution is the prior distribution and the belief of the user?s true long-term behavior trends is the posterior distribution. The goal is to detect a typical bot?s behavior of a narrow posterior distribution towards one extreme of the rating range or the other since this behavior is vastly different from a typical normal distribution or even a newbie legitimate user.

Model

The BIRDNEST model has two phases: fitting the Bayesian model (BIRD) to a given data set and creating the metric of suspiciousness called the Normalized Expected Surprise Total (NEST). It expects as input data the following: a matrix of users? ratings (rating from 1 to n) where the user is indexed from 1 to m and the corresponding matrix of timestamp of the rating. The timestamp data is pre-processed to calculate the IATs of each user.

The BIRD generative model is a mixture model where there are K clusters. Each of these clusters are trained by the user training data to represent various normal user prior distributions using the Dirichlet prior distribution. Once the mixture model converges, the posteriors are calculated based on the Dirichlet distribution property.

The NEST part of the algorithm uses the BIRD mixture model posterior distributions to calculate the suspiciousness of a user. The metric surprise is defined as the negative log of the global mixture model distribution given an observed user rating distribution p (similarly for temporal distribution q). Because rating and time have different ranges, the surprise metric of rating and time are normalized and summed for the final NEST value. The smaller the NEST value, the less suspicious the user is and it is less likely that the user is a bot.

Algorithm

The algorithm first maximizes the overall likelihood function to calculate the best posterior marginals. Then, the NEST metric is calculated to quantitatively determine how certain the model is of the user being fradulent.

1. Determine the number of clusters, K, by the Bayesian Information Criterion (BIC).
2. Optimize the mixture model for the given data set by iteratively adjusting the hyperparameters and recalculating the mixture weights until convergence
3. Calculate the posterior distributions of p_i and q_i using the conjugate prior property of Dirichlet distributions.
4. Calculate the surprise for rating and IAT distribution, normalize distributions by the standard deviation, and sum to create the NEST metric of suspiciousness.

While BIRDNEST is able to compute the degree of suspiciousness of a user rather than simply classifying it as a human or a bot, it assumes that a user's rating events are independent of each other (i.i.d. fallacy).

Group Latent Anomaly Detection (GLAD) – 2014

Approach

In this paper, the authors? goal is to determine group anomalies by proposing the GLAD (Group Latent Anomaly Detection) model, which uses a hierarchical Bayes model. Unlike previous models, GLAD takes in pair-wise and point-wise data as input and determines the groups and group anomalies simultaneously. Until now, people were only trying to detect individual anomalies; however, detecting group anomalies early and efficiently is also important in order for preventing group attacks. There are three main observations made towards group anomaly:

1. Both point-wise data and pairwise data exist in social media
2. Group anomaly is harder to detect than individual anomaly
3. Individuals are dynamic which makes detecting group anomaly harder to find

The authors first look at previous models that were used to detect group anomaly such as Multinomial Genre Model (MGM), Latent Dirichlet Allocation (LDA), Flexible Genre Model (FGM), and support measure machine (SMM). Both MGM and LDA define a group as a mixture of Gaussian distributed topics and gives each group a certain mixture rate. In order to determine the role mixture rate for each individual, we infer their role identity and group membership. If this mixture rate is significantly different than a mixture rate of normal groups, then we can classify this group as an anomaly. All of these models first determine the groups and then discover anomalies in a two-step process, but GLAD attempts to do this process simultaneously. However, all of these models, including GLAD, are only able to handle static data, which means that there needs to be another model which can handle both dynamic and temporal data.

Model

In general, the GLAD model first identifies the normal mixture rates and then observes the likelihood of the observations given these normal mixture rates. A particular group is an anomaly if this likelihood value is low.

Each person can belong to a group given a membership distribution which is defined by the Dirichlet prior. The multinomial distribution calculates the likelihood of that individual belonging to a specific group. Two people can be linked based on their group identities. Each group has a role mixture rate and each person can have multiple roles and multiple memberships. A limitation of GLAD is that it can only determine group anomaly if the groups stay static.

A modified version of the GLAD model, d-GLAD, can handle both dynamic data and detect mixture rates with respect to time with these groups. This model uses a variational Bayesian method and a Monte Carlo sampling technique. Essentially, at a certain time, a GLAD model is applied and the mixture rates are stored. This is done with the multivariate Gaussian distributions. Each mixture rate at a certain time is determined from the Gaussian distribution of the previous time mixture rate. When there is a significant change to the mixture rates, then there is an anomaly.

Both GLAD and d-GLAD were tested on synthetic and real data. In a dataset of scientific publications, GLAD was able to detect anomalous papers; in a dataset containing US Senate Voting, d-GLAD was able to detect changes in party affiliation over time. However, there are certain limitations such as group anomaly is difficult to detect without labeled data and the d-GLAD model is computationally expensive and is not scalable to a large dataset.

ND-SYNC – 2015

Approach

The authors first survey different forms of fraud-detection on Twitter, citing two common tactics that are explored in several other papers: simple feature-based detection and collective anomaly detection. ND-SYNC is the first to be textual content-agnostic, graph structure-agnostic, user attributes-agnostic, parameter-free, unsupervised, specifically designed for RT-Fraud problem, and a well-performing detector for synchronicity fraud.

Model

This paper investigates and proposes a method for detecting Twitter fraud as it pertains to fraudulent retweet activity (RTFraud). By analyzing numerous features captured over large retweet threads, they define and measure of the concept of synchronicity across multiple features. The authors tackle this by formulating RTFraud as a special case of the synchronicity fraud problem (SyncFraud). They extract features for each retweet thread and assign a suspiciousness score based on the similarities to other threads, assuming that organic behavior is the

norm.

Features

The *synchronicity* of a group is a measure of closeness between all of its members. The *normality* of a group with respect to a superset is a measure of closeness between members of the group and members of the superset. Finally, they define the residual score as the difference between the synchronicity and mathematically-defined lower bound on synchronicity. According to the SyncFraud problem, suspicious groups are those that exhibit highly synchronized characteristics.

To tackle the RTFraud problem specifically, the authors experimented with a large number of features and determined the seven most significant features to be:

- Number of retweets
- Response time of first retweet
- Lifespan, constrained to three weeks
- RT-Q3, the time to garner the first ? of the retweets
- RT-Q2, the time to garner the first half of the retweets
- Arr-MAD, mean absolute deviation of IATs
- Arr-IQR, inter-quartile range of IATs

By projecting their data onto two of these seven features, the authors came up with visualizations of their data that helped them find intuitive patterns in the data, namely lifespan vs. Arr-MAD and retweets vs. response time.

Algorithm

One of the major components in determining suspiciousness across all features is to do a sweep over all feature subspaces. Given p features, there are 2^p subspaces to examine, and the original p-dimensional entities must be projected onto each of them. The researchers show that even 3-D and 2-D subspaces are effective in building an intuition of the relationship between features.

The suspiciousness of a group with respect to a feature subspace is determined as the residual score of the projection of that group on that subspace. Intuitively, if the group is very synchronized in a given subspace, the group has a high suspiciousness score for that subspace. This is calculated for each group, for each of the 2^p subspaces. Unlike other algorithms, which usually detect outlying groups based on low inter-synchronicity, ND-Sync looks for high intra-synchronicity, which the researchers argue is a better indicator, as they find that that normal users are approximately at around the same level of inter-synchronicity. Essentially, real data has no "norm" for the data to fit, so detecting outliers proves fruitless. Instead, this approach looks for structure.

Discussion

aaa

Temporal Approach

aaa

Bayesian Approach

aaa

Group Approach

aaa