Probabilistic Inference on Twitter Data to Discover Suspicious Users and Malicious Content

Praveen Rao, Anas Katib
School of Computing & Engineering
University of Missouri-Kansas City
raopr@umkc.edu,anaskatib@mail.umkc.edu

Charles Kamhoua, Kevin Kwiat, and Laurent Njilla
Air Force Research Lab
Cyber Assurance Branch
{charles.kamhoua.1,kevin.kwiat,laurent.njilla}@us.af.mil

***Abstract*—While the power of social media on the Internet is undeniable, it has become a major weapon for launching cyberattacks against an organization and its people. Today, there is a growing number of cyberattacks being launched through social media such as posting of false content from hacked accounts, posting malicious URLs to spread malware, and others. In this paper, we present a simple and flexible unified framework called SocialKB for modeling social media posts and reasoning about them to ascertain their veracity, a first step towards discovering emerging cyber threats. SocialKB is based on Markov Logic Networks (MLNs), a popular representation in statistical relational learning. It learns a knowledge base (KB) on the social media posts and users' behavior in a unified manner. By conducting probabilistic inference on the KB, SocialKB can identify suspicious users and malicious content. In this work, we specifically focus on tweets posted by users on Twitter. Finally, we report an evaluation of SocialKB on 20,000 tweets and discuss our early inference results.**

I. Introduction

The power of social media is undeniable: may it be in a marketing or political campaign, sharing breaking news, or during catastrophic events. Unfortunately, social media has also become a major weapon for launching cyberattacks on an organization and its people.[1] By hacking into accounts of (popular) users, hackers can post false information, which can go viral and lead to economic damages and create havoc among people. For instance, a false tweet posted about the Syrian President's death using a news agency's Twitter feed caused oil prices to spike.[1] In 2015, it was reported that the Twitter accounts of U.S. military officials were hacked by those claiming to support the Islamic State.[2] Later that year, it was reported that Iranian hackers took control of social media accounts of U.S. State Department officials.[3] Another major threat on social media is the spread of malware through social media posts by tricking innocent users to click unsuspecting links [27]. Due to these reasons, organizations are developing policies for usage of social media and investing a lot of money and resources to secure their infrastructure and prevent such attacks.

Social media datasets can be regarded as "big data" due to their massive, complex, heterogeneous nature. Gartner reported that by 2016, 25% of large global companies will rely on big data analytics for cybersecurity and fraud.[4] While statistical models provide an elegant framework for gaining knowledge from complex datasets [17], the volume, velocity, variety, and veracity of big data demands a paradigm shift. Several big data ecosystems have been developed to tackle the volume, velocity, and variety of big data and gain insights from data. The veracity (or trustworthiness) of big data (*e.g.*, social media content) is becoming more important today [20, 1, 14]. To ascertain veracity of social media content, we must consider both the content as well as users' behavior.

In this paper, we present SocialKB, a simple and flexible unified framework for modeling and reasoning about the veracity of social media posts to discover suspicious users and malicious content. SocialKB builds on concepts in statistical relational learning for knowledge representation and reasoning under uncertainty. It can be used to analyze both the behavior of users and the nature of their posts to ultimately discover potential cyberattacks on social media. The nature of cyberattacks on social media is quite complex: It can range from posting of malicious URLs to spread malware, to posting of misleading/false information to create chaos, to compromise of innocent users' accounts. SocialKB learns a KB over social media data–to capture both the behavior of users and the nature of their posts. The KB will contain entities, their relationships, facts, and rules. Using the KB, one can efficiently reason about the veracity of the social media posts and users' behavior to flag suspicious content/activity in a timely manner. *To the best of our knowledge, none of the previous efforts has leveraged statistical relational learning techniques, specifically MLNs, for discovering cyber threats on social media.*

There are several technical challenges that arise in designing a unified framework such as SocialKB for social media data. The first challenge is to represent the complex and diverse social media data in a principled manner. For example, a tweet on Twitter is represented using 100+ attributes, and attribute values can be missing and noisy. New attributes may appear in tweets; some attributes may not appear in a tweet. Hashtags (*e.g.*, #baseball, #malware, #election2016) are used frequently

Approved for Public Release; Distribution Unlimited: 88ABW-2016 4368 20160907

[1] https://www.cisco.com/web/offer/gist_ty2_asset/Cisco_2013_ASR.pdf

[2] www.cnbc.com/2015/01/12/us-central-command-twitter-hacked.html

[3] www.nytimes.com/2015/11/25/world/middleeast/iran-hackers-cyberespionage-state-department-social-media.html

[4] www.gartner.com/doc/2651118

978-1-5090-4314-9/16 $25.00 © 2016 IEEE
DOI 10.1109/CIT.2016.29

by users in tweets to indicate specific topics or categories. There are thousands of hashtags in use today; the popularity of a hashtag changes over time. Some hashtags may become trending/popular during a particular time period. The second challenge is to construct a KB on millions of social media posts in a scalable and efficient manner. The goal is to learn the entities, facts, and rules from large number of posts. This is a key challenge because popular social media sites have millions of active users. For example, as of March 2016, Twitter had 310 million active users and receives 400 million tweets per day.[5] The designed framework should be able to grow the KB as new posts are produced. The third challenge is to reason about the veracity of the posts by performing fast probabilistic inference on the KB containing millions of entities and facts. Thus, suspicious content/activities can be flagged as soon as possible to discover emerging cyber threats. SocialKB attempts to address some of the above challenges.

The rest of the paper is organized as follows. Section II provides the background on MLNs and discusses related work on social media data analysis. Section III presents the novel design of our framework. Section IV reports an evaluation of our framework on tweets collected from Twitter. Section V provides some perspectives on how to extend our framework. Finally, we conclude in Section VI.

II. Background and Related Work

A. Markov Logic Network

In statistical relational learning, a MLN [21] is regarded as one of the most flexible representations as it combines first-order logic and probabilistic graphical models. MLNs are widely used in natural language processing [19], entity resolution [15, 25], hypertext classification [3], and information extraction [18] and retrieval. Formally, a MLN is a KB defined by a set of pairs (F,w), where F is a first-order formula that denotes a constraint and w is a real-valued weight of the formula. Higher the weight, more likely is the constraint believed to be satisfied in the set of possible worlds. A formula with infinite weight is a hard constraint. Formulas can contradict. A world that violates a formula is less probable but not impossible. However, a world that violates a hard constraint has zero probability. Once the formulas and weights are learned [24], probabilistic inference can be performed on the MLN by posing maximum a posteriori (MAP) and marginal inference queries. Efficient inferencing techniques have been developed (*e.g.*, lifted inference [9, 23]) including those that can operate on large KBs with millions of entities and facts by leveraging the scalability and efficiency of relational database systems and cluster computing (*e.g.*, Tuffy [16], ProbKB [5]).

Example 1: Consider a KB on smokers and friends with 3 formulas [21]: $\forall x\ Smoker(x) \implies Cancer(x)$; $\forall x \forall y\ Friends(x,y) \implies (Smoker(x) \iff Smoker(y))$; and $\forall x\ Smoker(x)$ with weights 3.5, 1.0, and -1.0, respectively. The first formula is a stronger constraint than the others and is of higher importance in the set of possible worlds. The third formula with -ve weight implies that a person is more likely to be a non-smoker. Using probabilistic inference, one can perform marginal inference queries on an entity or all entities (*e.g.*, $\Pr(Friends(Alice, Bob))$, $\Pr(Cancer(x))$) as well as MAP queries (*e.g.*, $\arg\max_x \Pr(Cancer(x))$).
A grounding of a formula (or predicate) is obtained by replacing all its variables by constants. The obtained formula (or predicate) is called a ground formula (or ground predicate). Given a MLN, its ground Markov network is denoted by (X, G), where X is the set of binary random variables and G is the set of ground formulas. For each ground predicate, there is one binary random variable in X. The set of possible worlds $\mathcal{X}$ is the set of all possible assignments of truth values to variables in X. The probability of a possible world is given by $\Pr(X = x) = \frac{1}{Z} \exp(\sum_i w_i n_i(x))$, where w_i is the weight of the i^{th} formula, $n_i(x)$ is the number of true groundings of the i^{th} formula in x, and $Z = \sum_{x' \in \mathcal{X}} \exp(\sum_i w_i n_i(x'))$ is a normalization constant.

B. Analysis of Content and User Behavior on Social Media

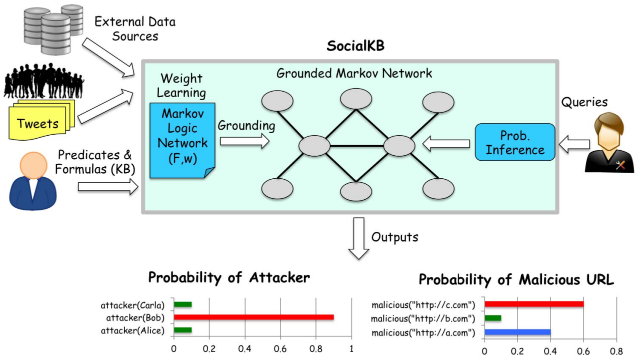
Several researchers have developed methods for detecting spam and propagation of malware on social media. Haberman *et al.* [8] was one of the early researchers to study the social interactions of users on Twitter based on their followers and followee counts. Lee *et al.* [11] proposed a honeypot-based approach to detect spammers on social media sites such as MySpace and Twitter. Stringhini *et al.* [26] studied the problem of detecting spammers (*e.g.*, single bots) and spam campaigns on three different social media sites using "honey-profiles."

Jin *et al.* [10] proposed a scalable spam detection system using data mining techniques. They considered both image and text content as well as social network features. Wang *et al.* [28] analyzed the misuse of shortened URLs on Twitter and classified them into spam and non-spam based on the click traffic data. Yang *et al.* [31] analyzed Twitter accounts and inner social relationships of users to detect criminal accounts. They concluded that criminal accounts tend to be socially connected forming a small-world network. Yan *et al.* [30] analyzed the social structure, click probability, and user activity patterns of an online social network called BrightKite to understand the characteristics of malware propagation.

Ghosh *et al.* [6] studied the problem of link farming on Twitter, wherein spammers attempt to acquire a large number of followers. They proposed methods to dissuade users from connecting to other users to increase their influence in a social network. Sanzgiri *et al.* [22] developed and studied three different attack models to analyze the spread of malware on Twitter. They showed that even with low degree of connectivity with users and low probability of users clicking the posted links, an attacker can infect several users. Lee *et al.* [12] developed a near-real time detection for suspicious URLs on Twitter. They identified the correlation of URL redirect chains in the tweets using frequently shared URLs

[5] https://en.wikipedia.org/wiki/Twitter

Fig. 1. The architecture of SocialKB



and eventually, training a classifier to detect suspicious URLs. Recently, Burnap *et al.* [2] developed a classification system using machine learning to detect malicious URLs soon after the links are clicked.

Researchers have also developed techniques to assess the veracity of social media content. Qazvinian *et al.* [20] studied the problem of rumor detection in microblogs using statistical methods. They used different kinds of features (*e.g.*, content-based, network-based) to correctly classify tweets that contain misinformation. Bodnar *et al.* [1] developed a veracity assessment model on users' social network profiles using natural language processing and machine learning techniques to classify information and misinformation about events on Twitter. Lukasik [14] studied the problem of classifying tweet level judgments of rumors on Twitter using supervised learning such as multi-task learning.

Unlike prior work, our goal is to develop a unified framework for modeling and reasoning about the veracity of social media posts to discover suspicious content/activity early on. Our key idea, which is explained in detail in the next section, is to leverage MLNs to construct a KB over social media content – to analyze both the behavior of users and the nature of their posts within the same framework – and reason over the constructed KB via probabilistic inference.

III. Our Framework

In this section, we present SocialKB, a unified framework for modeling and reasoning about the veracity of social media posts. In addition, we discuss how we tackle some of the technical challenges presented in Section I.

A. *Overview of SocialKB*

The overall architecture of SocialKB is shown in Figure 1. A domain expert/user defines the input KB containing predicates and first-order formulas over social media content. Once the predicates are known, we automatically generate the evidence dataset, which is a set of ground predicates observed by combining social media content with external data sources. An example of an external data source is URLBlacklist.com[6], which provides a categorization of URLs and domains (*e.g.*, whitelist, spyware, virus infected, adult, violence). The current version of SocialKB is built using Tuffy [16], which leverages a relational database system to perform weight learning and probabilistic inference efficiently. SocialKB learns the weights of the formulas in the KB on the evidence dataset containing millions of entities and ground predicates using Tuffy. (It is also possible for SocialKB to discover new rules from the KB using ProbKB [5].) When an end-user poses inference queries such as MAP and marginal inference queries, SocialKB performs probabilistic inference (again using Tuffy) on the learned MLN and outputs the ground predicates that are satisfied/true. Essentially, an end-user can use SocialKB to identify suspicious users, malicious URLs, sensitive tweets, and so on. The ultimate goal of SocialKB is to enable end-users to reason about the veracity of the social media posts to flag suspicious content/activity early on.

B. *Modeling Tweets using a KB*

In this work, we focus on social media content posted by users on Twitter. Using public APIs provided by Twitter, we can collect a large number of public tweets as well as other information such as trending hashtags, followers of users, friends of users, etc. We begin by discussing how we model tweets using a KB in a principled manner.

1) Attributes and Content in a Tweet: Tweets are rich in information and diverse in the sense that they may contain 100+ attributes, and new attributes may appear over time. Each tweet is assigned a unique ID; each user account is also assigned a unique ID. In subsequent discussions, we simply say "a user" to mean "a user account." There are attributes whose values embed the actual text of a tweet, the URLs contained in a tweet, hashtags used in a tweet, and so on. There are attributes that provide counts about the friends, followers, and statuses (*i.e.*, number of posts) of a user. Note that a tweet does not contain the list of friends/followers of a user. Nor does it contain information about trending hashtags. These pieces of information, however, can be obtained using Twitter APIs.[7]

2) Predicates in the KB: Due on the richness of information in tweets and complex relationships between entities in them, we define a set of different types of predicates in the KB. A predicate can make a closed-world assumption (CWA) or an open-world assumption (OWA). CWA assumes that what is not known to be true must be false. On the other hand, OWA assumes that what is not known may or may not be true.

Figure 2(a) shows the first set of predicates based on CWA. The predicate *tweeted(userID,tweetID)* states whether a user posted a particular tweet or not; *containsLink(tweetID,link)* states whether a tweet contains a particular URL or not; *containsHashtag(tweetID,hashtag)* states whether a tweet contains a particular hashtag or not; *mentions(tweetID,userID)* states whether a particular user is mentioned in a tweet (using the @ symbol) or not; *retweeted(userID,tweetID)* states whether a user retweeted a particular tweet or not; finally, *verified(userID)* states whether a user has been verified or

[6] http://urlblacklist.com

[7] https://dev.twitter.com/rest/public

Fig. 2. Set of predicates in the KB based on CWA (denoted by *) and OWA

*tweeted(userID,tweetID)
*containsLink(tweetID,link)
*mentions(tweetID,userID)
*retweeted(userID,tweetID)
*containsHashtag(tweetID,hashtag)
*verified(userID)
(a)

malicious(link)
friend(userID1,userID2)
trending(hashtag)
attacker(userID)
isFollowedBy(userID1,userID2)
isPossiblySensitive(tweetID)
(b)

*friendsCount(userID,count)
*followersCount(userID,count)
*statusesCount(userID,count)
*retweetCount(tweetID,count)
*favouritesCount(userID,count)
(c)

*tweetedT(userID,tweetID,t)
trendingT(hashtag,t)
*followersCountT(userID,count,t)
*friendsCountT(userID,count,t)
(d)

Fig. 3. First set of first-order formulas in the KB

$\mathbf{f_1}$: tweeted(userID1,tweetID) $\wedge$ mentions(tweetID,userID2) $\Longrightarrow$ friend(userID1,userID2)
$\mathbf{f_2}$: retweeted(userID1,tweetID) $\wedge$ tweeted(userID2,tweetID) $\Longrightarrow$ friend(userID1,userID2)
$\mathbf{f_3}$: tweeted(userID,tweetID) $\wedge$ containsHashtag(tweetID,hashtag) $\wedge$ attacker(userID) $\Longrightarrow$ trending(hashtag)
$\mathbf{f_4}$: isFollowedBy(userID1,userID2) $\wedge$ verified(userID2) $\Longrightarrow$ verified(userID1)

Fig. 4. Second set of first-order formulas in the KB

$\mathbf{f_5}$: verified(userID) $\Longrightarrow$!attacker(userID)
$\mathbf{f_6}$: verified(userID1) $\wedge$ friend(userID1,userID2) $\wedge$ isFollowedBy(userID1,userID2) $\Longrightarrow$!attacker(userID2)
$\mathbf{f_7}$: tweeted(userID,tweetID) $\wedge$ containsLink(tweetID,link) $\wedge$ malicious(link) $\Longrightarrow$ attacker(userID)
$\mathbf{f_8}$: attacker(userID1) $\wedge$ friend(userID1,userID2) $\wedge$ isFollowedBy(userID1,userID2) $\Longrightarrow$ attacker(userID2)
$\mathbf{f_9}$: !attacker(userID1) $\wedge$ tweeted(userID1,tweetID) $\wedge$ mentions(tweetID,userID2) $\Longrightarrow$!attacker(userID2)
$\mathbf{f_{10}}$: tweeted(userID,tweetID) $\wedge$ isPossiblySensitive(tweetID) $\Longrightarrow$ attacker(userID)

not. Twitter independently verifies user accounts that are of public interest in domains such as government, fashion, music, politics, sports, etc.

Figure 2(b) shows the next set of predicates based on OWA. The predicate *malicious(link)* states whether a URL is malicious or not; *friend(userID1,userID2)* states whether a user denoted by *userID1* has a friend denoted by *userID2* or not. Twitter defines a friend as someone who a user is following. The predicate *trending(hashtag)* indicates if a hashtag is trending or not; *attacker(userID)* indicates whether a user is a suspicious user or not; *isFollowedBy(userID1, userID2)* indicates whether a user denoted by *userID1* is followed by another user denoted by *userID2* or not; and finally, *isPossiblySensitive(tweetID)* indicates whether a tweet is possibly sensitive or not. Twitter flags a tweet as possibly sensitive based on users' feedback.

To model the count information in a tweet, we define a set of predicates as shown in Figure 2(c). These predicates model the friends count/followers count/statuses count of a user, the retweet count of a tweet, and the number of tweets a user has "liked." These predicates are based on a CWA. For instance, if a user denoted by *userID* has a friend count of 100, then *friendsCount(userID,count)* is true only when count equals 100 and false for all other values of count.

The predicates described thus far do not contain temporal information. One interesting aspect of using a MLN to model tweets is that we can define predicates with temporal variables. These predicates are shown in Figure 2(d). The predicate *tweetedT(userID, tweetID, t)* indicates whether a user posted a particular tweet at a particular time or not; *trendingT(hashtag, t)* indicates whether a hashtag is trending at a particular time or not; *followersCountT(userID, count, t)* indicates whether a user has a particular followers count at a particular time or not; and finally, *friendsCount(userID, count, t)* indicates whether a user has a particular friends count at a particular time or not. These predicates specify temporal constraints on users' behavior on social media.

3) First-Order Formulas in the KB: At the core of SocialKB is a set of constraints/first-order formulas defined on the predicates. In the current version of SocialKB, these formulas were constructed based on the findings in published literature [31, 28, 22, 2], observing our personal account activities on Twitter, and through intuitive reasoning. These formulas can contradict each other. Each formula will be assigned a weight, which can be learned over a training dataset. A world that violates a formula is less probable but not impossible. A formula with a +ve weight is more likely to be true in the set of possible worlds; a formula with a -ve weight is less likely to be true. A world that violates a hard constraint (assigned the weight ∞) has zero probability.

Next, we introduce the first-order formulas in SocialKB. The existential quantifier $\exists$ on each variable in a formula is implied. The first set of formulas shown in Figure 3 infers friendship relations, trending hashtags, and verified users. Formula $\mathbf{f_1}$ states that if a user mentions another user in his/her tweet, then this implies that the mentioned user is a friend of the user. Formula $\mathbf{f_2}$ states that if a user retweets a tweet of another user, then the friend relationship between the two users is implied. Formula $\mathbf{f_3}$ states that if a user posted a hashtag and is an attacker/suspicious user, then this implies that the hashtag is trending as adversaries are more likely to target trending hashtags. Formula $\mathbf{f_4}$ states that if a user is followed by a verified user, then this implies that the user is also verified/trustworthy.

The second set of formulas shown in Figure 4 infers whether a user is an attacker/suspicious user or not. Formula $\mathbf{f_5}$ states that if a user is verified, then he/she is not an attacker; formula $\mathbf{f_6}$ states that a friend of a verified user is not an attacker; formula $\mathbf{f_7}$ states that a user who posted a tweet containing a

Fig. 5. Third set of first-order formulas in the KB

$\mathbf{f_{11}}$: containsLink(tweetID,link) $\wedge$ [contains(link,ϕ)] $\implies$!malicious(link) // hard constraint with wt. ∞
$\mathbf{f_{12}}$: containsLink(tweetID,link) $\wedge$ isPossiblySensitive(tweetID) $\implies$ malicious(link)
$\mathbf{f_{13}}$: attacker(userID) $\wedge$ tweeted(userID,tweetID) $\wedge$ containsLink(tweetID,link) $\implies$ malicious(link)
$\mathbf{f_{14}}$: containsLink(tweetID,link) $\wedge$ malicious(link) $\implies$ isPossiblySensitive(tweetID)
$\mathbf{f_{15}}$: attacker(userID) $\wedge$ tweeted(userID,tweetID) $\implies$ isPossiblySensitive(tweetID)

Fig. 6. Fourth set of first-order formulas on predicates denoting counts

$\mathbf{f_{16}}$: !verified(user) $\wedge$ followersCount(user,count1) $\wedge$ friendsCount(user, count2) $\wedge$ [count1 != 0 AND count2/count1 $> m$] $\implies$ attacker(user)
$\mathbf{f_{17}}$: !verified(user) $\wedge$ statusesCount(user,count1) $\wedge$ friendsCount(user,count2) $\wedge$ [count1 != 0 AND count2/count1 $> m$] $\implies$ attacker(user)
$\mathbf{f_{18}}$: !verified(user) $\wedge$ statusesCount(user,count1) $\wedge$ followersCount(user,count2) $\wedge$ [count1 != 0 AND count2/count1 $> m$] $\implies$ attacker(user)
$\mathbf{f_{19}}$: !verified(user) $\wedge$ statusesCount(user,count1) $\wedge$ favouritesCount(user,count2) $\wedge$ [count1 != 0 AND count2/count1 $> m$] $\implies$ attacker(user)

malicious link is an attacker; formula $\mathbf{f_8}$ states that a friend of an attacker is also an attacker; formula $\mathbf{f_9}$ states that if a user, who is not an attacker, mentions another user in his/her tweet, then the other user is not an attacker; and finally, formula $\mathbf{f_{10}}$ states that if a user's tweet is known to be possibly sensitive, then he/she is an attacker.

The third set of formulas shown in Figure 5 infers whether a link is malicious or not and whether a tweet is possibly sensitive or not. Formula $\mathbf{f_{11}}$ states that a URL containing a particular prefix ϕ is not malicious. We define this formula as a hard constraint. The prefix can be https://t.co/, which indicates the use of Twitter's URL shortening service. Or it could denote trusted domains such as https://twitter.com, https://www.instagram, http://pinterest.com, etc. Formula $\mathbf{f_{12}}$ states that a URL contained in a possibly sensitive tweet is malicious; formula $\mathbf{f_{13}}$ states that a URL in a tweet posted by an attacker is malicious; formula $\mathbf{f_{14}}$ states that a tweet containing a malicious URL is possibly sensitive; and finally, formula $\mathbf{f_{15}}$ states that a tweet of an attacker is possibly sensitive.

The fourth set of formulas shown in Figure 6 infers attackers based on the counts of certain attributes in the tweets. In these formulas, m denotes a positive integer constant. Formula $\mathbf{f_{16}}$ states that if a non-verified user has a very large number of users he/she is following compared to the number of users following him/her, then the user is an attacker. Formulas $\mathbf{f_{17}}$, $\mathbf{f_{18}}$, and $\mathbf{f_{19}}$ state that if a non-verified user is not active on Twitter (based on the number of posts) but has a large number of friends/followers or has liked a large number of tweets, then the user is an attacker. Note that when a user's tweet is liked by someone, then a notification is sent to the user. Thus, a suspicious user can drawn the attention of other users to himself/herself by randomly liking their tweets. Similarly, a user can mention any other user in his/her tweet to seek attention.

Fig. 7. Fifth set of first-order formulas on predicates with temporal variables

$\mathbf{f_{20}}$: friendsCountT(user,count1,t1) $\wedge$ friendsCountT(user,count2,t2) $\wedge$ [t1 - t2 <= τ AND count1 != 0 AND count2/count1 $> m$] $\implies$ attacker(user)
$\mathbf{f_{21}}$: trendingT(hashtag,t1) $\wedge$ tweetedT(user,tweet,t2) $\wedge$ containsHashtag(tweet,hashtag) $\wedge$ containsLink(tweet,link) $\wedge$ attacker(user) $\wedge$ [t1 $<$ t2] $\implies$ malicious(link)
$\mathbf{f_{22}}$: trendingT(hashtag,t1) $\wedge$ tweetedT(user1,tweet,t2) $\wedge$ mentions(tweet,user2) $\wedge$!isFollowedBy(user2,user1) $\wedge$ [t1 $<$ t2] $\implies$ attacker(user1)
$\mathbf{f_{23}}$: trendingT(hashtag,t1) $\wedge$ tweetedT(user1,tweet,t2) $\wedge$ mentions(tweet,user2) $\wedge$!friend(user2,user1) $\wedge$ [t1 $<$ t2] $\implies$ attacker(user1)

The last set of formulas in the KB is defined over predicates with temporal variables. (See Figure 7.) These formulas are powerful to model a sequence of activities, which can be exploited by adversaries to launch cyberattacks. Formula $\mathbf{f_{20}}$ states that if the friends count of a user (*i.e.*, the number of users being followed by the user) increases substantially during a predefined time interval τ, then the user is a suspicious user as he/she is trying to increase their social influence. Formula $\mathbf{f_{21}}$ states that if a hashtag is trending at a point in time, and an attacker posts a tweet containing that hashtag a later time, and if the tweet contains a URL, then it is implied to be malicious. This constraint enables us to capture the actions of an attacker who is tracking trending hashtags to post malicious URLs to maximize the scope of an attack. Formulas $\mathbf{f_{22}}$ and $\mathbf{f_{23}}$ state that if a hashtag is trending at a point in time, and a user posts a tweet containing that hashtag at a later time, and mentions another user who he/she is not following or is not friends with, then the user is an attacker. This constraint allows us to model attackers who can mention other users in their posts randomly just to have malicious content sent to those innocent users.

We would like reiterate that some of the formulas in the KB can be thought of as untested hypotheses. Their true effect on the results of probabilistic inference will depend on their actual weights, which can be learned from the evidence dataset.

IV. Evaluation

In this section, we present an evaluation of SocialKB on tweets collected from Twitter. We report early results to show that SocialKB holds promise in discovering suspicious users and malicious content posted on social media.

A. Software, implementation, and hardware setup

We collected tweets using Apache Spark's stream processing APIs[8] and Twitter4J[9]. We downloaded the URL and domain datasets published by http://URLBlacklist.com. This

[8] http://spark.apache.org/docs/latest/streaming-programming-guide.html
[9] http://twitter4j.org

TABLE I
EVIDENCE DATASET: FREQUENCY & TYPE OF GROUND PREDICATES

Predicate	CWA	Count
tweeted(userID,tweetID)	Yes	27,902
containsLink(tweetID,link)	Yes	5,049
containsHashtag(tweetID,hashtag)	Yes	6,356
mentions(tweetID,userID)	Yes	15,891
retweeted(userID,tweetID)	Yes	8,530
verified(userID)	Yes	64
friendsCount(userID,count)	Yes	19,623
retweetCount(tweetID,count)	Yes	8,363
followersCount(userID,count)	Yes	19,629
statusesCount(userID,count)	Yes	19,931
favouritesCount(userID,count)	Yes	0
tweetedT(userID,tweetID,t)	Yes	0
followersCountT(userID,count,t)	Yes	0
friendsCountT(userID,count,t)	Yes	0
malicious(link)	No	21
friend(userID1,userID2)	No	157,297
trending(hashtag)	No	3,421
attacker(userID)	No	0
isFollowedBy(userID1,userID2)	No	12,438,943
isPossiblySensitive(tweetID)	No	179
trendingT(hashtag,t)	No	0
Total	-	12,731,199

website provides regularly updated categorization of URLs and domains into 70+ categories, including whitelist (suitable for children), adult, violence, spyware, virus infected, and others. We used Spark SQL to process the collected tweets and mark URLs in them as malicious using the collected datasets. Specifically, we considered the hacking, malware, spyware, phishing, and virus infected categories to be malicious. We used Tuffy for learning the weights of the MLN in SocialKB and conducting probabilistic inference on the KB. Tuffy exploits a relational database system (*i.e.*, PostgreSQL) in a novel way for efficient MLN weight learning and inference queries over large datasets. It is written in Java and runs on a single machine using multiple threads to speed up the execution.

We ran all the experiments on a 64-bit Ubuntu 12.04 machine with 6 Intel Xeon 1.6 GHz cores and 32 GB RAM. We used a 2 TB hard disk to store tweets, the KB, and temporary data of Tuffy.

B. Tweets and the Evidence Dataset

We collected 20,000 tweets from Twitter during June 2016. We did not specify any filters during the data collection. As expected, each tweet had over 100 attributes. We created a set of ground predicates for these tweets. For trending hashtags, we used Twitter's REST APIs to collect them. For a subset of verified users, we also collected their friendship lists and their followers as this information was not explicitly available in the tweets. In total, the evidence dataset had 12,731,199 million ground predicates. Table I shows the frequency and type of ground predicates in the evidence dataset. Note that the predicate *tweeted(userID,tweetID)* is more than 20,000 because some tweets had other tweets embedded in them due to retweeting as well as quoting of others' tweets. We did not generate certain ground predicates in the evidence dataset (*e.g.*, in Figure 2(d)) and report a count of 0 in the table.

C. KB Construction

Given the predicates, first-order formulas, and the evidence dataset, we used Tuffy to learn the weights for the formulas. Tuffy uses the discriminative weight learning approach proposed by Lowd *et al.* [13]. Tuffy required 14 hours and 2 minutes to learn the weights. Table II shows the weights learned by Tuffy for different formulas. (Hard constraints are not shown here and have ∞ as the weight. Recall that a possible world that violates a hard constraint has zero probability.) The number of formulas in Table II is smaller than the original set as we selected three queries of interest for probabilistic inference: *attacker(u)*, *malicious(l)*, and *isPossiblySensitive(t)*. During weight learning, Tuffy considered only those formulas that affect the queries considered for inference. It is interesting to note that some of the formulas, which we believed would have +ve weights, turned out to have -ve weights after weight learning. This shows that our formulas could be conflicting, but weight learning will rely on the evidence dataset to construct the appropriate weights.

D. MAP Inference Queries

Next, we present the results of MAP inference using the constructed KB on the evidence dataset. Note that MAP inference computes the most likely possible world. The output of the MAP inference task contained ground predicates (corresponding to the queries) that are most likely to be satisfied/true. We executed three queries: *attacker(u)*, *malicious(l)*, and *isPossiblySensitive(t)*. The MAP inference task took 4 hours and 42 minutes on the evidence dataset.

Let us first analyze the results of the MAP inference task for *malicious(l)*. Note that only 21 URLs were known to be malicious in the evidence dataset. For those newly identified URLs, which were not in the evidence dataset, we used VirusTotal[10], a free online service that aggregates the output/analysis of several antivirus engines and website scanners on suspicious URLs and files. For each URL that was output by the MAP inference task, we first checked if that URL was flagged as malicious by at least one antivirus engine/scanner using the public APIs of VirusTotal. If not, we obtained the IP address for the URL's domain, and then fetched an aggregated report for that IP address from VirusTotal. The report contained passive DNS information on the IP address as well as latest URLs hosted in this address and detected as malicious by at least one URL scanner or malicious URL dataset. As shown in Table III, out of 84 URLs identified by the MAP inference task, 4 were flagged by VirusTotal as malicious by directly searching for the URL, and 66 were flagged as malicious after resolving the IP address for the domain of the URL. For 14 of the URLs, we could not find any malicious reports on VirusTotal, which we refer to as benign.

[10] http://www.virustotal.com

TABLE II
WEIGHTS LEARNED BY TUFFY FOR THE RELEVANT FORMULAS IN SOCIALKB

$\mathbf{f_1}$	$\mathbf{f_3}$	$\mathbf{f_5}$	$\mathbf{f_6}$	$\mathbf{f_7}$	$\mathbf{f_8}$	$\mathbf{f_9}$	$\mathbf{f_{10}}$	$\mathbf{f_{12}}$	$\mathbf{f_{13}}$	$\mathbf{f_{14}}$	$\mathbf{f_{15}}$	$\mathbf{f_{17}}$	$\mathbf{f_{18}}$
-9.906	20.213	14.754	16.175	5.387	17.150	21.189	5.042	5.433	20.031	3.703	21.750	-9.826	-11.839

TABLE III
RESULTS FOR THE INFERENCE QUERY *malicious(l)*

Type of inference	# of URLs identified (total)	Malicious		Benign (% of total)
		Direct URL search	After IP address resolution	
MAP	84	4	66	14 (16.6%)
Marginal	72	3	57	12 (16.6%)

TABLE IV
RESULTS FOR THE INFERENCE QUERY *attacker(u)*

Type of inference	# of users identified	# of user accounts suspended	# of users who tweeted malicious URLs
MAP	496	11	413
Marginal	450	7	392

TABLE V
RESULTS FOR THE INFERENCE QUERY *isPossiblySensitive(t)*

Type	# of tweets identified	Tweets containing malicious URLs		
		URL Blacklist .com	Direct URL search	After IP address resolution
MAP	366	22	13	313
Marginal	306	18	9	278

It is promising to observe that only 16.6% of URLs output by SocialKB were incorrectly detected as malicious.

Next, let us discuss the results of the MAP inference task for *attacker(u)*. As shown in Table IV, 496 users were identified as suspicious users/attackers. Of these, 11 user accounts were suspended by Twitter or removed. Among the identified users, 413 of them had posted tweets that contained malicious URLs, which included those in the evidence dataset and the ones identified by VirusTotal.

Finally, for *isPossiblySensitive(t)*, the results are shown in Table V. Of the 366 newly identified tweets, 95% of the tweets contained malicious URLs: 22 of the tweets had URLs that were malicious according to http://URLBlacklist.com, and 326 of the tweets had URLs that were flagged as malicious by VirusTotal. Once again, this shows the effectiveness of SocialKB.

E. Marginal Inference Queries

We also performed the marginal inference task for the same set of queries. This task completed in 4 hours and 8 minutes. Marginal inference outputs ground predicates (of the queries) along with their probabilities of being true/satisfied. We further analyzed the ground predicates output whose probability was 0.8 or higher.

First, we present the results of the marginal inference task for *malicious(l)*. We processed the URLs the same way as we did before for MAP inference using VirusTotal. As shown in Table III, out of 72 URLs identified by marginal inference with probability 0.8 or higher, 3 were flagged by VirusTotal as malicious by directly searching for the URL, and 57 were flagged as malicious after resolving the IP address for the domain of the URL. For 12 of the URLs, we could not find any malicious reports on VirusTotal, *i.e.*, benign. As before, only 16.6% of URLs output by SocialKB were incorrectly detected as malicious.

Next, let us discuss the results of the marginal inference task for *attacker(u)*. As shown in Table IV, 450 users were identified as suspicious users/attackers with probability 0.8 or higher. Of these, 7 user accounts were suspended by Twitter or removed. Among the identified users, 392 users had posted tweets that contained malicious URLs, which included those in the evidence dataset and the ones identified by VirusTotal.

Finally, for *isPossiblySensitive(t)*, the results are shown in Table V. Of the 306 tweets newly identified as being possibly sensitive through the marginal inference task (with probability 0.8 or higher), 305 tweets contained malicious URLs according to either URLBlacklist.com or VirusTotal.

Note that a majority of the ground predicates output by marginal inference were in the probability range (0.8-1.0]. For malicious URLs, the percentage of ground predicates in this probability range was 77%. For suspicious users, the percentage of ground predicates in the same range was 89%. Finally, for possibly sensitive tweets, the percentage was 92%.

V. DISCUSSION

SocialKB is promising for discovering suspicious users and malicious content on Twitter. Our ultimate goal is to learn the KB (*i.e.*, formulas and their weights) and conduct probabilistic inference on millions of tweets, *i.e.*, big data. Note that the current implementation of SocialKB uses Tuffy. While Tuffy performs well on millions of ground predicates using multiple cores on a single machine, ProbKB [5] is reported to perform better than Tuffy on even larger datasets using a shared-nothing cluster. Thus on larger evidence datasets, ProbKB should be used in SocialKB.

In this work, we manually constructed the first-order formulas in the KB. (The weights were learned by Tuffy over the evidence dataset.) We do not think this is a serious barrier for using MLNs. Using recently proposed techniques [29, 4], SocialKB can potentially learn new rules automatically over social media posts. This way SocialKB can perform inference on a richer KB leading to improved accuracy.

In the context of adversarial machine learning [7], which is an important topic of research in security, we must analyze different attack models that are possible in SocialKB.

SocialKB does not assume that the data is independently and identically distributed (i.i.d.). This is because MLNs can model non-i.i.d. data. For a causative attack, where the adversary manipulates the training process by influencing the training data, it is more challenging to do so in SocialKB, because we can potentially learn the weights for the formulas on millions of ground predicates in the evidence dataset. The adversary has to generate a very large number of tweets compared to existing users on Twitter to influence the weight learning for the formulas. If the formulas are kept secret, then it becomes even harder for the adversary to guess them as there will be an exponential number of possibilities (w.r.t. the number of predicates) in the KB. We acknowledge that a thorough analysis of the attack models in the context of adversarial machine learning is needed.

VI. Conclusion

In this paper, we made the case for a simple and flexible unified framework called SocialKB to model complex social media posts and reason over it to ascertain the veracity of the posts. Specifically, we focused on tweets from Twitter. The primary goal of SocialKB is to identify suspicious users and malicious content early on to discover emerging cyber threats on social media. SocialKB is novel in the sense that it leverages MLNs to model a variety of complex predicates and constraints using first-order logic. In addition to non-temporal predicates and constraints, we can also express those that contain temporal attributes. We presented promising results on how SocialKB can be used to flag suspicious users and malicious content on 20,000 tweets.

Acknowledgments

The first author was supported by the U.S. Air Force Summer Faculty Fellowship and the National Research Council Research Associateship Senior Fellowship Award.

References

[1] T. Bodnar, C. Tucker, K. Hopkinson, and S. G. Bilen. Increasing the Veracity of Event Detection on Social Media Networks through User Trust Modeling. In *Proc. of IEEE International Conference on Big Data*, pages 636–643, Oct 2014.

[2] P. Burnap, A. Javed, O. F. Rana, and M. S. Awan. Real-time Classification of Malicious URLs on Twitter Using Machine Activity Data. In *Proc. of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015*, pages 970–977, 2015.

[3] S. Chakrabarti, B. Dom, and P. Indyk. Enhanced Hypertext Categorization Using Hyperlinks. In *Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data*, pages 307–318, Seattle, Washington, USA, 1998.

[4] Y. Chen, S. Goldberg, D. Z. Wang, and S. S. Johri. Ontological pathfinding. In *Proc. of the 2016 SIGMOD Conference*, pages 835–846, San Francisco, California, USA, 2016.

[5] Y. Chen and D. Z. Wang. Knowledge Expansion over Probabilistic Knowledge Bases. In *Proc. of the 2014 ACM SIGMOD Conference*, pages 649–660, 2014.

[6] S. Ghosh, B. Viswanath, F. Kooti, N. K. Sharma, G. Korlam, F. Benevenuto, N. Ganguly, and K. P. Gummadi. Understanding and Combating Link Farming in the Twitter Social Network. In *Proc. of the 21st International Conference on World Wide Web*, pages 61–70, 2012.

[7] L. Huang, A. D. Joseph, B. Nelson, B. I. Rubinstein, and J. D. Tygar. Adversarial Machine Learning. In *Proceedings of the 4th ACM Workshop on Security and Artificial Intelligence*, pages 43–58, Chicago, Illinois, USA, 2011.

[8] B. Huberman, D. Romero, and F. Wu. Social Networks that Matter: Twitter Under the Microscope. *First Monday*, 14(1), 2008.

[9] A. K. Jha, V. Gogate, A. Meliou, and D. Suciu. Lifted Inference Seen from the Other Side: The Tractable Features. In *Proc. of Advances in Neural Information Processing Systems (NIPS)*, pages 973–981, 2010.

[10] X. Jin, C. X. Lin, J. Luo, and J. Han. SocialSpamGuard: A Data Mining-Based Spam Detection System for Social Media Networks. *Proc. of the VLDB 2011*, 4(12):1458–1461, 2011.

[11] K. Lee, J. Caverlee, and S. Webb. Uncovering Social Spammers: Social Honeypots + Machine Learning. In *Proc. of the 33rd International SIGIR Conference*, pages 435–442, 2010.

[12] S. Lee and J. Kim. WarningBird: A Near Real-Time Detection System for Suspicious URLs in Twitter Stream. *IEEE Transactions on Dependable and Secure Computing*, 10(3):183–195, 2013.

[13] D. Lowd and P. Domingos. Efficient Weight Learning for Markov Logic Networks. In *Proc. of the 11th European Conference on Principles and Practice of Knowledge Discovery in Databases*, pages 200–211, 2007.

[14] M. Lukasik, T. Cohn, and K. Bontcheva. Classifying Tweet Level Judgements of Rumours in Social Media. In *Proc. of Empirical Methods in Natural Language Processing Conference*, pages 2590–2595, 2015.

[15] A. Mccallum and B. Wellner. Conditional Models of Identity Uncertainty with Application to Noun Coreference. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 905–912. MIT Press, Cambridge, MA, 2004.

[16] F. Niu, C. Ré, A. Doan, and J. Shavlik. Tuffy: Scaling Up Statistical Inference in Markov Logic Networks Using an RDBMS. *Proc. VLDB Endow.*, 4(6):373–384, Mar. 2011.

[17] NRC. *Frontiers in Massive Data Analysis*. The National Academies Press, Washington, DC, 2013.

[18] H. Poon and P. Domingos. Joint Inference in Information Extraction. In *Proceedings of the 22Nd National Conference on Artificial Intelligence - Volume 1*, pages 913–918, Vancouver, British Columbia, Canada, 2007.

[19] H. Poon and P. Domingos. Joint unsupervised coreference resolution with markov logic. In *Proceedings of the Conf. on Empirical Methods in NLP*, pages 650–659, 2008.

[20] V. Qazvinian, E. Rosengren, D. R. Radev, and Q. Mei. Rumor Has It: Identifying Misinformation in Microblogs. In *Proc. of the Conf. on Empirical Methods in NLP*, pages 1589–1599, 2011.

[21] M. Richardson and P. Domingos. Markov Logic Networks. *Machine Learning*, 62(1-2):107–136, Feb. 2006.

[22] A. Sanzgiri, A. Hughes, and S. Upadhyaya. Analysis of Malware Propagation in Twitter. In *Proc. of the 32nd IEEE Symposium on Reliable Distributed Systems*, pages 195–204, 2013.

[23] S. Sarkhel, P. Singla, and V. Gogate. Fast Lifted MAP Inference via Partitioning. In *Proc. of Advances in Neural Information Processing Systems (NIPS)*, pages 3240–3248, 2015.

[24] P. Singla and P. Domingos. Discriminative Training of Markov Logic Networks. In *Proc. of the 20th AAAI Conference on Artificial Intelligence*, pages 868–873, 2005.

[25] P. Singla and P. Domingos. Entity Resolution with Markov Logic. In *Proceedings of the Sixth International Conference on Data Mining*, ICDM '06, pages 572–582, 2006.

[26] G. Stringhini, C. Kruegel, and G. Vigna. Detecting Spammers on Social Networks. In *Proc. of the 26th Annual Computer Security Applications Conference*, pages 1–9, 2010.

[27] K. Thomas and D. M. Nicol. The Koobface botnet and the rise of social malware. In *Proc. of the 5th International Conference on Malicious and Unwanted Software*, pages 63–70, Oct 2010.

[28] D. Wang, S. B. Navathe, L. Liu, D. Irani, A. Tamersoy, and C. Pu. Click Traffic Analysis of Short URL Spam on Twitter. In *Proc. of 9th International Conference on Collaborative Computing: Networking, Applications and Worksharing*, pages 250–259, Oct 2013.

[29] W. Y. Wang, K. Mazaitis, and W. W. Cohen. Structure learning via parameter learning. In *Proc. of the 23rd CIKM Conference*, pages 1199–1208, Shanghai, China, 2014.

[30] G. Yan, G. Chen, S. Eidenbenz, and N. Li. Malware Propagation in Online Social Networks: Nature, Dynamics, and Defense Implications. In *Proc. of the 6th ACM Symposium on Information, Computer and Communications Security*, pages 196–206, 2011.

[31] C. Yang, R. Harkreader, J. Zhang, S. Shin, and G. Gu. Analyzing Spammers' Social Networks for Fun and Profit: A Case Study of Cyber Criminal Ecosystem on Twitter. In *Proc. of the 21st International Conference on the World Wide Web*, pages 71–80, 2012.