HoloScope: Topology-and-Spike Aware Fraud Detection

Shenghua Liu,[1,2,*] Bryan Hooi,[2] Christos Faloutsos[2*]
[1]CAS Key Laboratory of Network Data Science & Technology,
Institute of Computing Technology, Chinese Academy of Sciences
[2]Carnegie Mellon University
liu.shengh@gmail.com,{bhooi,christos}@cs.cmu.edu

ABSTRACT

As online fraudsters invest more resources, including purchasing large pools of fake user accounts and dedicated IPs, fraudulent attacks become less obvious and their detection becomes increasingly challenging. Existing approaches such as average degree maximization suffer from the bias of including more nodes than necessary, resulting in lower accuracy and increased need for manual verification. Hence, we propose HoloScope, which introduces a novel metric "contrast suspiciousness" integrating information from graph topology and spikes to more accurately detect fraudulent users and objects. Contrast suspiciousness dynamically emphasizes the contrast patterns between fraudsters and normal users, making HoloScope capable of distinguishing the synchronized and anomalous behaviors of fraudsters on topology, bursts and drops, and rating scores. In addition, we provide theoretical bounds for how much this increases the time cost needed for fraudsters to conduct adversarial attacks. Moreover, HoloScope has a concise framework and sub-quadratic time complexity, making the algorithm reproducible and scalable. Extensive experiments showed that HoloScope achieved significant accuracy improvements on synthetic and real data, compared with state-of-the-art fraud detection methods.

CCS CONCEPTS

• **Information systems → Data mining**; • **Computing methodologies → Anomaly detection**;

KEYWORDS

Graph Mining, Time Series, Fraud Detection, Burst and Drop

1 INTRODUCTION

Online fraud has become an increasingly serious problem due to the high profit it offers to fraudsters, which can be as much as $5 million from 300 million fake "views" per day, according to a report of Methbot [28] on Dec 2016. Meanwhile, to avoid detection, fraudsters can manipulate their geolocation, internet providers, and IP address, via large IP pools (852,992 dedicated IPs). Suppose a fraudster has a accounts or IPs, and serves a customer who buys 200 ratings or clicks for each of his products. Since the fraudster has to add 200 ratings to each product out of a possible a ratings, the density of the fraudulent block created is $200/a$. We thus see that with enough user accounts or IPs, the fraudster can serve as many products as he needs while keeping density low. This presents a difficult challenge for most existing fraud detection methods.

Due to the lack of labeled data in fraud detection, unlike email spam detection, many studies on fraud detection use unsupervised approaches, i.e. dense block detection. Current dense block detection methods [5, 35, 36] maximize the arithmetic or geometric average degree. We use "fraudulent density" to indicate the edge density that fraudsters create for target objects. However, those methods have a bias of including more nodes than necessary, especially as the fraudulent density decreases, which we verified empirically. This bias results in low precision, which then requires intensive manual work to verify each user. Fraudar [13] proposed an edge weighting scheme based on inverse logarithm of objects' degrees to reduce this bias, which was inspired by IDF [32, 37]. However, their weighting scheme is fixed globally and affects both suspicious and normal edges, lowering the precision of Fraudar, which can be seen from results on semi-real (with injected labels) and real data (see Fig. 1).

Accurately detecting fraudulent blocks of lower density requires aggregating more sources of information [12, 14, 35]. Consider the attribute of the creation time of edges: fraudulent attacks tend to be concentrated in time, e.g., fraudsters may surge to retweet a message, creating one or more sudden bursts of activity [9, 40], followed by sudden drops after the attack is complete. Sudden bursts and drops have not been directly considered together in previous work.

Therefore, we propose HoloScope, an unsupervised approach, which combines suspicious signals from graph topology, temporal bursts and drops, and rating deviation. Our graph topology-based weighting scheme dynamically reweights objects according to our beliefs about which users are suspicious. Temporally, HoloScope detects suspicious spikes of bursts and drops, which increases the time cost needed for fraudsters to conduct an attack. In terms of rating deviation, our approach takes into account how much difference there is between an object's ratings given by suspicious users and non-suspicious users.

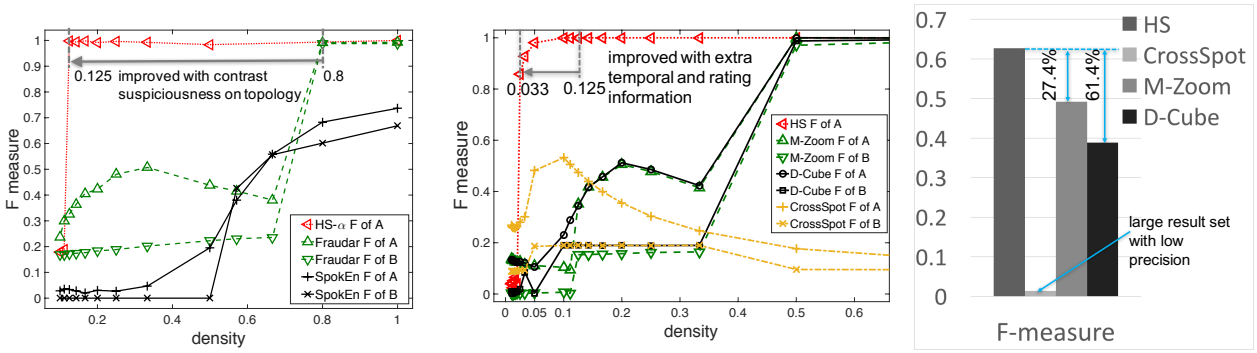
In summary, our contributions are:

- **Novel suspiciousness metric:** we propose a dynamic *contrast suspiciousness* metric, which emphasizes the contrast behaviors between fraudsters and honest users in an unsupervised way. At the same time, the contrast suspiciousness provides a unified suspiciousness framework, which can

*The work was done when Shenghua Liu was a visiting researcher at Carnegie Mellon University

CIKM'17, , November 6–10, 2017, Singapore.

ACM ISBN 978-1-4503-4918-5/17/11…$15.00
https://doi.org/10.1145/3132847.3133018



(a) HS-α using topology information outperforms baselines

(b) HS using holistic attributes provides clear improvement, and performs the best.

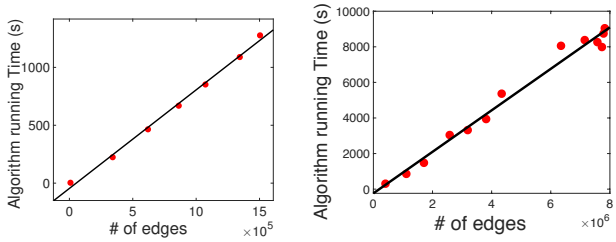
(c) HS achieves the best F-measure, on real data from Sina Weibo

Figure 1: (a) and (b) show experimental results on BeerAdvocate dataset. The better methods are able to detect fraud with high accuracy, even when fraudulent density (plotted on the horizontal axis) is low. HS-α and HS are both our methods, where the former only uses topology information. We increase the # of injected fraudsters from 200 to 2000 for HS-α, and to 20000 for HS, while the decreasing density of fraudulent edges is shown on the horizontal axis from right to left. Comparing with HS-α, HS who makes holistic use of several signals achieves further improvement. (c) shows accuracy (F measure of precision and recall) results on Sina Weibo, with ground truth labels.

make holistic use of several signals including, but not limited to, connectivity (i.e., topology), temporal bursts and drops, and rating deviation in a systematic way.

- **Robustness and theoretical analysis of fraudsters' obstruction:** we show that if the fraudsters use less than a theoretical bound of time for an attack, they will cause a suspicious drop or burst. In other words, HoloScope obstructs fraudsters by increasing the time they need to perform an attack. This theorem guarantees temporal robustness: no matter how the fraudsters manipulate the creation time of fraudulent links, they will be caught if the attack takes less than a fixed amount of time.
- **Effectiveness:** we achieved higher accuracy than the baselines on semi-real and real datasets. In fact, HoloScope using only topology information (HS-α) outperformed the graph-based baselines (see Fig. 1a), while HoloScope (HS) using all signals achieved further improvement, and outperformed the tensor-based baselines (see Fig. 1b and 1c). The dynamic weighting on object nodes by contrast suspiciousness makes both HS-α and HS resistant to fraudsters' camouflage and achieve better detection accuracy.
- **Scalability:** HoloScope runs in subquadratic time in the number of nodes, under reasonable assumptions. Fig. 2 shows that its running time increases near-linearly with the number of edges.

In addition, in Microblog, Sina Weibo[1] data, HoloScope achieved higher F-measure than the baselines in detecting the ground truth labels, with high precision and recall. The code of HoloScope is open-sourced for reproducibility [2].



(a) BeerAdvocate dataset

(b) Amazon Electronics dataset

Figure 2: HoloScope (HS) runs in near-linear time.

Table 1: Comparison between HoloScope and other fraud detection algorithms.

	Fraudar [13]	SpokEn [31]	CopyCatch [4]	CrossSpot [14]	BP-based methods [1, 29]	M-Zoom/D-Cube [35, 36]	HoloScope
scalability	✓	✓	✓	✓	✓	✓	✓
camouflage	✓			✓		✓	✓
hy-community			?	?			✓
spike-aware			?				✓

2 RELATED WORKS

Most existing works study fraud detection in an unsupervised way due to the limited labels, which are based on the density of blocks within adjacency matrices [18, 31], or multi-way tensors [35, 36]. OddBall [2] found new rules and patterns in the distribution of eigenvalues for anomaly detection. In stead of detecting density block by average degree [5], [10] and CoreScope [34] proposed to

[1] The largest Microblog service in China, http://www.weibo.com
[2] https://github.com/shenghua-liu/HoloScope

use Shingling and K-core algorithms respectively to detect anomalous dense block in huge graphs. Taking into account the suspiciousness of each edge or node in a real life graph potentially allows for more accurate detection. Fraudar [13] proposed to weight edges' suspiciousness by the inverse logarithm of objects' indegrees, to discount popular objects. [3] found that the degrees in a large community follow a power law distribution, forming hyperbolic structures. This suggests penalizing high degree objects to avoid unnecessarily detecting the dense core of hyperbolic community. The spikes on degree distributions were studied, and synchronized behaviors were detected in [16, 17]. Deep neural network methods are used for anomaly detection [20, 23], but these are black-box approaches that provide little interpretability about the detected output.

In addition to topological density, EdgeCentric [33] studied the distribution of rating scores to find the anomaly. In terms of temporal attribute, the identification of burst period has been studied in [21]. A recent work, Sleep Beauty (SB) [19] more intuitively defined the awakening time for a paper's citation burst for burst period. [39] detected the outliers of time series as the changing point. [41] clustered the temporal patterns of text phrases and hash tags in Twitter, and [27, 30] studied the temporal dynamics of networks separately on ego-network and network motifs. Meanwhile, [11, 12] modeled the time stamped rating scores with Bayesian model and autoregression model respectively for anomalous behavior detection. Even though [7, 22, 40] have used bursty patterns to detect review spam, a sudden drop in temporal spikes has not been considered yet. [40] detected spam reviews in singleton reviews, where each spammer only writes one or fewer reviews in the system. The algorithms found the common period in which multiple time series have bursts, including the time series of ratios of singleton reviewers. We solve a different fraud detection problem that spammers have to reuse the limited accounts to create as many fake reviews as possible, which needs to find suspicious signals from topological connections at the same time.

Aggregating suspiciousness signals from different attributes is challenging for unsupervised learning. [6] proposed RRF (Reciprocal Rank Fusion) scores for combining different rank lists in information retrieval. However, RRF applies to ranks, not suspiciousness scores. Without explicitly aggregation, researchers used the tensor-based methods to consider different attributes. CrossSpot [14, 15], a tensor-based algorithm, estimated the suspiciousness of a block using a Poisson model. However, it did not take into account the difference between popular and unpopular objects. Moreover, although CrossSpot, M-Zoom [35] and D-Cube [36] can consider edge attributes like rating time and scores via a multi-way tensor approach, they require a time-binning approach. When time is split into bins, attacks which create bursts and drops may not stand out clearly after time-binning, since each time bin is treated as an independent dimension in the temporal way of tensor. The problem of choosing bin widths for histograms was studied by Sturges [38] assuming an approximately normal distribution, and Freedman-Diaconis [8] based on statistical dispersion. However, the binning approaches were proposed for the time series of a single object, which is not clear for different kinds of objects in a real life graph, namely, popular products and unpopular products should use different bin sizes.

Belief propagation (BP) [29] is another common approach for fraud detection which can be incorporated some specific edge attributes, such as sentiments [1]. However, its robustness against adversaries which try to hide themselves is not well understood. Based on the same idea of BP, CopyCatch [4] detected lockstep behavior by maximizing the number of edges in blocks constrained within time windows. However, this approach ignores the distribution of edge creation times within the window, and does not capture bursts and drops directly.

Finally, we summarize the previous baselines compared to our HoloScope in Table 1. We use "hy-community" to indicate whether the method can avoid detecting the naturally-formed hyperbolic topology that is unnecessary (false positive) for fraud detection. We can see that HoloScope is the only one which considers all the property list, especially including temporal spikes (sudden bursts and drops, and multiple bursts) and hyperbolic topology, in a unified suspiciousness framework.

3 PROPOSED APPROACH

The definition of our problem is as follows.

Problem 1 (informal definition). ***Given*** *quadruplets (user, object, timestamp, #stars), where timestamp is the time that a user rates an object, and #stars is the categorical rating scores.*

- ***Find*** *a group of suspicious users, and suspicious objects or its rank list with suspiciousness scores,*
- ***to optimize*** *the metric under the common knowledge of suspiciousness from topology, rating time and scores.*

To make the problem more general, *timestamp* and *#stars* are optional. For example, in Twitter, we have $(user, object, timestamp)$ triples, where *user* retweets a message *object* at *timestamp*. In a static following network, we have pairs $(user, object)$, with *user* following *object*.

As discussed in previous sections, our metric should capture the following basic traits.

First, the fraudsters need to create as many fake reviews as they can to boost fraudulent products.

Trait 1 (Engagement). *Fraudsters engage as much firepower as possible to boost customers' objects, i.e., suspicious objects.*

Second, as [13] suggested, a popular object by many people is not likely a fraudulent object. In other words, suspicious objects attract less attention from ordinary users due to their low quality. Then we have:

Trait 2 (Less Involvement). *Suspicious objects seldom attract non-fraudulent users to connect with them.*

Third, fraudsters conduct their attacks in a short period of time, creating temporal spikes with bursts and sudden drops, as reported in previous works [9, 40].

Trait 3 (Spikes: Bursts and Drops). *Fraudulent attacks are concentrated in time, sometimes over multiple waves of attacks, creating bursts of activity. Conversely, the end of an attack corresponds to sudden drops in activity.*

Finally, the rating distribution of fraudsters differs greatly from those of typical users, as observed by [33]. This occurs because fraudsters are aiming to manipulate the rating of products.

Trait 4 (Rating Deviation). *The rating behavior of fraudsters deviates greatly from the rating behavior of normal users.*

Thus we will show in the following sections, that our proposed metric can make holistic use of several signals, namely topology, temporal spikes, and rating deviation, to locate suspicious users and objects satisfying the above traits. That is the reason we name our method as HoloScope (HS).

3.1 HoloScope metric

To give a formal definition of our metric, we describe the quadruplets (*user*, *object*, *timestamp*, *#stars*) as a bipartite and directed graph $\mathcal{G} = \{U, V, E\}$, which U is the source node set, V is the sink node set, and connections E is the directed edges from U to V. Generally, graph $\mathcal{G}$ is a multigraph, i.e., multiple edges can be present between two nodes. Multiple edges mean that a user can repeatedly comment or rate on the same product at a different time, as common in practice. Users can also retweet message multiple times in the Microblog *Sina Weibo*. Each edge can be associated with rating scores (*#stars*), and timestamp, for which the data structure is introduced in Subsection 3.1.2.

Our HoloScope metric detects fraud from three perspectives: topology connection, timestamp, and rating score. To easily understand the framework, we first introduce the HoloScope in a perspective of topology connection. Afterwards, we show how we aggregate the other two perspectives into the HoloScope. We first view $\mathcal{G}$ as a weighted adjacency matrix $\mathbb{M}$, with the number of multiple edges (i.e., edge frequency) as matrix elements.

Our goal is to find lockstep behavior of a group of suspicious source nodes $A \subset U$ who act on a group of sink nodes $B \subset V$. Based on Trait 1, the total engagement of source nodes A to sink nodes B can be basically measured via density measures. There are many density measures, such as arithmetic and geometric average degree. Our HoloScope metric allows for any such measure. However, as the average degree metrics have a bias toward including too many nodes, we use a measure denoted by $D(A, B)$ as the basis of the HoloScope, defined as:

$$D(A, B) = \frac{\sum_{v_i \in B} f_A(v_i)}{|A| + |B|} \tag{1}$$

where $f_A(v_i)$ is the total edge frequency from source nodes A to a sink node v_i. $f_A(v_i)$ can also be viewed as an engagement from A to v_i, or A's lockstep on v_i, which is defined as

$$f_A(v_i) = \sum_{(u_j, v_i) \in E \wedge u_j \in A} \sigma_{ji} \cdot e_{ji} \tag{2}$$

where constant σ_{ji} is the global suspiciousness on an edge, which can be equal to 1 if no extra global suspiciousness is assigned to a node pair (u_j, v_i). We are going to propose a way to assign the suspiciousness in section 3.1.2. e_{ji} is the element of adjacency matrix $\mathbb{M}$, i.e., the edge frequency between a node pair (u_j, v_i). The edge frequency e_{ji} becomes a binary in a simple graph. The global suspiciousness as a prior can come from the degree, and the extra knowledge on fraudsters, such as duplicated review sentences and unusual behaving time.

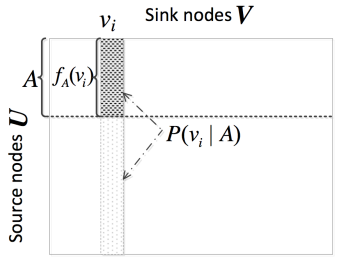


Figure 3: An intuitive view of our definitions in the HoloScope.

To maximize $D(A, B)$, the suspicious source nodes A and the suspicious sink nodes B are mutually dependent. Therefore, we introduce *contrast suspiciousness* in an informal definition:

Definition 3.1 (contrast suspiciousness). The contrast suspiciousness denoted as $P(v_i \in B|A)$ is defined as the conditional likelihood of a sink node v_i that belongs to B (the suspicious object set), given the suspicious source nodes A.

A visualization of the contrast suspiciousness is given in Fig. 3. The intuitive idea behind contrast suspiciousness is that in the most case, we need to judge the suspiciousness of objects by currently chosen suspicious users A, e.g., an object is more suspicious if very few users not in A are connected to it (see Trait 2); the sudden burst of an object is mainly caused by A (see Trait 3); or the rating scores from A to an object are quite different from other users (see Trait 4). Therefore, such suspiciousness makes use of the contrasts between users in A and users not in A or the whole set.

Finally, instead of maximizing $D(A, B)$, we maximize the following expectation of suspiciousness $D(A, B)$ over the probabilities $P(v_i \in B|A)$:

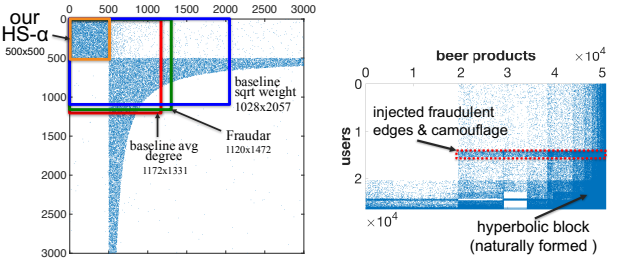
$$\begin{aligned} \max_A HS(A) &:= \mathbb{E}\left[D(A, B)\right] \\ &= \frac{1}{|A| + \sum_{k \in V} P(v_k|A)} \sum_{i \in V} f_A(v_i) P(v_i|A) \end{aligned} \tag{3}$$

where for simplicity we write $P(v_i|A)$ to mean $P(v_i \in B|A)$. $1 - P(v_i|A)$ is the probability of v_i being a normal sink node. We dynamically calculate the contrast suspiciousness for all the objects, after every choice of source nodes A.

Using this overall framework for our proposed metric $HS(A)$, we next show how to satisfy the remaining Traits. To do this, we define contrast suspiciousness $P(v_i|A)$ in a way that takes into account various edge attributes. This will allow greater accuracy particularly for detecting low-density blocks.

3.1.1 HS-α: Less involvement from others. Based on Trait 2, a sink node should be more suspicious if it only attracts connections from the suspicious source nodes A, and less from other nodes. Mathematically, we capture this by defining

$$P(v_i|A) \propto q(\alpha_i), \text{ where } \alpha_i = \frac{f_A(v_i)}{f_U(v_i)} \tag{4}$$



(a) HS-α finds the exact dense block in the synthetic data **(b) Hyperbolic community in BeerAdvocate data**

Figure 4: (a) The synthetic data consists of hyperbolic and rectangular blocks, with volume density around 0.84 and 0.60 respectively. The camouflage is randomly biased to columns of high degree. (b) A real data of naturally-formed hyperbolic community, and injected dense block. The injection is 2000×200 with biased camouflage.

where $f_U(v_i)$ is the weighted indegree of sink node v_i. Similar to $f_A(v_i)$, the edges are weighted by global suspiciousness. α_i measures the involvement ratio of A in the activity of sink node v_i. The scaling function $q(\cdot)$ is our belief about how this ratio relates to suspiciousness, and we choose the exponential form $q(x) = b^{x-1}$, where base $b > 1$.

As previous work showed, large communities form hyperbolic structures, which is generated in our synthetic data (see the lower-right block in Fig. 4a), and also exists in real BeerAdvocate data (see Fig. 4b). For clarity, our HoloScope method are denoted as HS-α when it is only applied on a connection graph. The results of the synthetic data show that HS-α detected the exact dense rectangular block ($b = 128$), while the other methods included a lot of non-suspicious nodes from the core part of hyperbolic community resulting in low accuracy. In the beer review data from the BeerAdvocate website, testing on different fraudulent density (see Fig. 1a), our HS-α remained at high accuracy, while the other methods' accuracy drops quickly when the density drops below 70%.

The main idea is that HS-α can do better because it dynamically adjusts the weights for sink nodes, penalizing those sink nodes that also have many connections from other source nodes not in A. In contrast, although Fraudar proposed to penalize popular sink nodes based their indegree, these penalties also scaled down the weights of suspicious edges. The Fraudar (green box) only improved the unweighted "average degree" method (red box) by a very limited amount. Moreover, with a heavier penalty, the "sqrt weight" method (blue box) achieved better accuracy on source nodes but worse accuracy on sink nodes, since those methods used globally fixed weights, and the weights of suspicious were penalized as well. Hence the hyperbolic structure pushes those methods to include more nodes from its core part.

In summary, our HS-α using dynamic contrast suspiciousness can improve the accuracy of fraud detection in 'noisy' graphs (containing hyperbolic communities), even with low fraudulent density.

3.1.2 Temporal bursts and drops. Timestamps for edge creation are commonly available in most real settings. If two subgroups of Microblog users have the same number of retweets to a message, can we say they have the same suspiciousness? As an example shown in Fig. 5a, the red line is the time series (histogram of time bins) of the total retweets of a message in Microblog, Sina Weibo. The blue dotted line and green dashed line are the retweeting time series respectively from user groups A_1 and A_2. The two series have the same area under the time series curves, i.e., the same number of retweets. However, considering that fraudsters tend to surge to retweet a message to reduce the time cost, the surge should create one or more sudden bursts, along with sudden drops. Therefore, the suspiciousness of user groups A_1 and A_2 become quite different even though they have the same number of retweets, which cannot be detected solely based on connections in the graph. Thus we include the temporal attribute into our HoloScope framework for defining contrast suspiciousness.

Denote the list of timestamps of edges connected to a sink node v as T_v. To simplify notation, we use T without subscript when talking about a single given sink node v. Let $\mathcal{T} = \{(t_0, c_0), (t_1, c_1), \cdots, (t_e, c_e)\}$ as the *time series* of T, i.e., the histogram of T. The count c_i is the number of timestamps in the time bin $[t_i - \Delta t/2, t_i + \Delta t/2)$, with bin size Δt. The bin size of histogram is calculated according to the maximum of Sturges criteria and the robust Freedman-Diaconis' criteria as mentioned in related works. It is worth noticing that the HoloScope can *tune* different bin sizes for different sink nodes, e.g., popular objects need fine-grained bins to explore detailed patterns. Hence, the HoloScope is more flexible than tensor based methods, which use a globally fixed bin size. Moreover, the HoloScope can update the time series at a low cost when T is increasing.

To consider the burst and drop patterns described in Trait 3, we need to decide the start point of a burst and the end point of a drop in time series $\mathcal{T}$. Let the burst point be (t_m, c_m), having the maximum value c_m. According to the definition in previous work "Sleeping Beauty", we use an auxiliary straight line from the beginning to the burst point to decide the start point, named the *awakening point* of the burst. Fig. 5b shows the time series $\mathcal{T}$ (red polygonal line) of a message from Sina Weibo, the auxiliary straight line l (black dotted line) from the lower left point (t_0, c_0) to upper right point (t_m, c_m), and the awakening point for the maximum point (t_m, c_m), which is defined as the point along the time series $\mathcal{T}$ which maximizes the distance to l. As the dotted line perpendicular to l suggests in this figure, the awakening point (t_a, c_a) satisfies

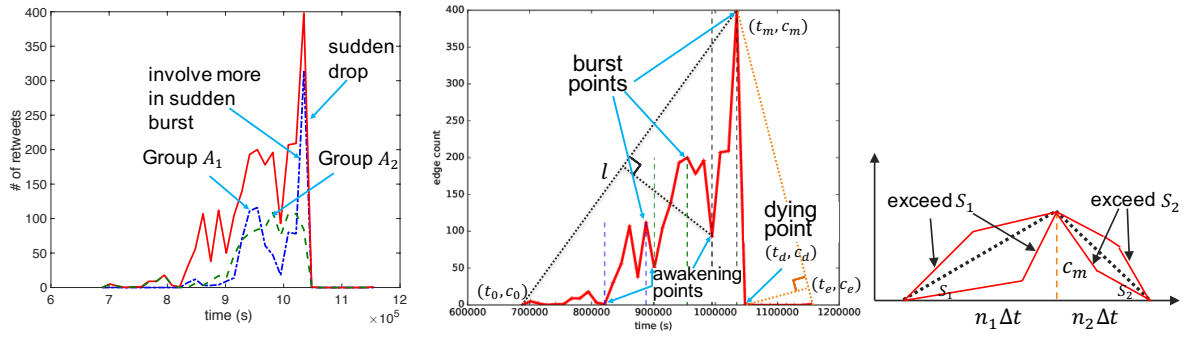
$$t_a = \underset{(c,t)\in\mathcal{T}, t<t_m}{\arg\max} \frac{|(c_m - c_0)t - (t_m - t_0)c + t_m c_0 - c_m t_0|}{\sqrt{(c_m - c_0)^2 + (t_m - t_0)^2}} \quad (5)$$

Finding the awakening point for one burst is not enough, as multiple bursts may be present. Thus, sub-burst points and the associated awakening points should be considered. We then propose a recursive algorithm *MultiBurst* in Alg. 1 for such a purpose.

After finding awakening and burst points, the contrast suspiciousness of burst awareness satisfies $P(v_i|A) \propto q(\varphi_i)$, where φ_i is the involvement ratio of source nodes in A in multiple bursts. Let the collection of timestamps from A to sink node v_i be T_A. Then,

$$\varphi_i = \frac{\Phi(T_A)}{\Phi(T_U)}, \text{ and } \Phi(T) = \sum_{(t_a,t_m)} \Delta c_{am} \cdot s_{am} \sum_{t\in T} \mathbf{1}(t \in [t_a, t_m]) \quad (6)$$

where s_{am} is the slope from the output of *MultiBurst* algorithm. Here s_{am} is used as a weight based on how steep the current burst



(a) Group A_1 is more suspicious than A_2, due to the sudden burst and drop

(b) Detection of temporal bursts and drops

(c) Proof the time cost obstruction

Figure 5: (a) and (b) are the time series (histogram) of a real message being retweeted in Microblog, Sina Weibo. The horizontal axis is the seconds after $2013-11-1$. (c) illustrates our proof of time cost obstruction.

Algorithm 1 *MultiBurst* algorithm.

Input Time series $\mathcal{T}$ of sink node v, beginning index i, end index j
Output A list of awakening-burst point pairs,
  s_{am}: slope of the line passing through each point pair,
  Δc: altitude difference of each point pair.
If $j - i < 2$ **then return**
(t_m, c_m) = point of maximum altitude between indices i and j.
(t_a, c_a) = the awakening point as Eq (5) between indices i and j.
$\Delta c_{am} = c_m - c_a$, and $s_{am} = {}^{\Delta c_{am}}/_{(t_m - t_a)}$
Append $\{(t_a, c_a), (t_m, c_m)\}$, s_{am}, and Δc_{am} into the output.
$MultiBurst(\mathcal{T}, i, a-1)$
k = Find the first local min position from indices $m+1$ to j
$MultiBurst(\mathcal{T}, k, j)$

is. This definition of suspiciousness satisfies Trait 3. It is worth noticing that the *MultiBurst* algorithm only needs to be executed once. With the preprocessed awakening and burst points, the contrast suspiciousness of edges connected to v has $O(d_v)$ complexity, where d_v is the degree of sink node v. Hence the complexity for overall sink nodes are $O(|E|)$.

In fact, sudden drops are also a prominent pattern of fraudulent behavior as described in Trait 3, since after creating the attack is complete, fraudsters usually stop their activity sharply. To make use of the suspicious pattern of a sudden drop, we define the *dying point* as the end of a drop. As Fig. 5b suggests, another auxiliary straight line is drawn from the highest point (t_m, c_m) to the last point (t_e, c_e). The dying point (t_d, c_d) can be found by maximizing the distance to this straight line. Thus we can discover the "sudden drop" by the absolute slope value $s_{bd} = {}^{(c_m - c_d)}/_{(t_d - t_m)}$ between the burst point and the dying point. Since there may be several drops in a fluctuated time series $\mathcal{T}$, we choose the drop with the maximum fall. To find the maximum fall, we also need a recursive algorithm, similar to Alg. 1:

1) Find a maximum point (t_m, c_m), and the corresponding dying point (t_d, c_d) by definition;
2) Calculate the current drop slope s_{bd}, and the drop fall $\Delta c_{bd} = c_m - c_d$;
3) Recursively find drop slope and drop fall for the left and right parts of $\mathcal{T}$, i.e., $t < t_m$ and $t \geq t_d$ respectively.

As a result, the algorithm returns the maximum drop fall Δc_{bd}, and its drop slope s_{bd}, which it has found recursively. Finally, we use the weighted drop slope as a global suspiciousness in equation (2) to measure the drop suspiciousness, i.e.

$$\sigma = \Delta c_{bd} \cdot s_{bd} \tag{7}$$

for every edge connected to the sink node v, and we omit the subscripts of nodes for simplicity.

With this approach to detect bursts and drops, we now show that this provides a provable time obstruction for fraudsters.

Theorem 3.2. *Let N be the number of edges that fraudsters want to create for an object. If the fraudsters use time less than $\tau \geq \sqrt{\frac{2N\Delta t(S_1+S_1)}{S_1 \cdot S_1}}$, then they will be tracked by a suspicious burst or drop, where Δt is the size of time bins, and S_1 and S_2 are the slopes of normal rise and decline respectively.*

Proof. The most efficient way to create N edges is to have one burst and one drop, otherwise more time is needed. As shown in Fig. 5c, in order to minimize the slope, every point in the time series should in line with the two auxiliary straight lines to the highest point c_m, separately from the awakening and dying points. Otherwise, the slopes will exceed the normal values S_1 and S_2. Hence we only consider the triangle with the auxiliary lines as its two edges. It is worth noticing that a trapezoid whose legs have the same slopes as the triangle's edges cannot have a shorter time cost. Then

$$\frac{c_m}{n_1\Delta t} = S_1, \qquad \frac{c_m}{n_2\Delta t} = S_2, \qquad (n_1+n_2)\cdot c_m = 2N'.$$

Here n_1 and n_2 are the number of time bins before and after the burst. N' is the total number of rating edges, and $N' \geq N$ consider some edges from normal users. Thus, solving the above equations, we have

$$\tau = (n_1+n_2)\Delta t = \sqrt{\frac{2N'\Delta t(S_1+S_2)}{S_1 \cdot S_2}} \geq \sqrt{\frac{2N\Delta t(S_1+S_2)}{S_1 \cdot S_2}}$$

■

We also have the height of burst, $c_m \geq \sqrt{\frac{2N\Delta t S_1 S_2}{S_1+S_2}}$. Thus, the maximum height of time series $\mathcal{T}$ cannot be larger by far than that of a normal sink node. That is the reason that we use the weighted φ_i in equation (6) and weighted drop slope in equation (2).

3.1.3 Rating deviation and aggregation. We now consider edges with categorical attributes such as rating scores, text contents, etc. For each sink node v_i, we use the KL-divergence κ_i between the distributions separately from the suspicious source nodes A and the other nodes, i.e., $U \setminus A$. We use $U \setminus A$ for KL-divergence instead of the whole source nodes U, in order to avoid the trivial case where most of the rating scores are from A. The rating deviation κ_i is scaled into $[0,1]$ by the maximum value before being passed into function $q(\cdot)$ to compute contrast suspiciousness. The neutral scores can be ignored in the KL-divergence for the purpose of detecting fraudulent boosting or defamation. Moreover, rating deviation is meaningful when both A and $U \setminus A$ have the comparable numbers of ratings. Thus, we weighted κ_i by a balance factor, $\min\{ {f_A(v_i)}/{f_{U\setminus A}(v_i)}, {f_{U\setminus A}(v_i)}/{f_A(v_i)} \}$.

To make holistic use of different signals, i.e., topology, temporal spikes, and rating deviation, we need a way to aggregate those signals together. As far as we know, there are few approaches that can be used for aggregation in an unsupervised framework. We have tried to use RRF (Reciprocal Rank Fusion) scores from Information Retrieval, and wrapped the scores with and without scaling function $q(x)$. Compared to RRF score, we found that a natural way of using joint probability was the most effective way to aggregate, i.e. multiplying those signals together:

$$P(v_i|A) = b^{\alpha_i+\varphi_i+\kappa_i-3}, \tag{8}$$

In such a way, we can consider the absolute suspicious value of each signal, as opposite to the only use of ranking order. Moreover, being wrapped with $q(x)$, the signal values cannot be canceled out by multiplying a very small value from other signals. A concrete example is that a suspicious spike can still keep a high suspiciousness score by multiplying a very small score from low fraudulent density.

Moreover, HoloScope dynamically updates the contrast suspiciousness $P(v_i|A)$. Thus the sink nodes being added with camouflage will have a very low contrast suspiciousness, with respect to the suspicious source nodes A. This offers HoloScope the resistance to camouflage.

3.2 Algorithm

Before designing the full algorithm for large scale datasets, we firstly introduce the most important sub-procedure *GreedyShaving* in Alg. 2.

At the beginning, this greedy shaving procedure starts with an initial set $A_0 \subset U$ as input. It then greedily deletes source nodes from A, according to users' scores $\mathcal{S}$:

$$\mathcal{S}(u_j \in A) = \sum_{v_i:(u_j,v_i)\in E} \sigma ji \cdot e_{ji} \cdot P(v_i|A),$$

which can be interpreted as how many suspicious nodes that user u_j is involved in. So the user is less suspicious if he has a smaller score, with respect to the current contrast suspiciousness $\mathcal{P}$, where we use $\mathcal{P}$ to denote a vector of contrast suspiciousness of all sink nodes. We build a priority tree to help us efficiently find the user with minimum score. The priority tree updates the users' scores and maintains the new minimum as the priorities change. With removing source nodes A, the contrast suspiciousness $\mathcal{P}$ change, in which we then update users' scores $\mathcal{S}$. The algorithm keeps reducing A until it is empty. The best A^* maximizing objective HS and $P(v|A^*)$ are returned at the end.

Algorithm 2 *GreedyShaving* Procedure.

Given bipartite multigraph $\mathcal{G}(U, V, E)$,
  initial source nodes $A_0 \subset U$.
Initialize:
  $A = A_0$
  $\mathcal{P}$= calculate contrast suspiciousness given A_0
$\mathcal{S}$ = calculate suspiciousness scores of source nodes A.
MT = build priority tree of A with scores $\mathcal{S}$.
while A is not empty **do**
  u = pop the source node of the minimum score from MT.
  $A = A \setminus u$, delete u from A.
  Update $\mathcal{P}$ with respect to new source nodes A.
  Update MT with respect to new $\mathcal{P}$.
  A^* = source nodes A that has the best objective $HS(A^*)$ so far.
end while
return A^* and $P(v|A^*)$, $v \in V$.

Since awakening and burst points have been already calculated for each sink node as an initial step before the *GreedyShaving* procedure, the calculation of the contrast suspicious $P(v|A)$ for a sink node v only needs $O(|A|)$ time. With source node j as the j-th one removed from A_0 by the *GreedyShaving* procedure, $|A_0| = m_0$, and the out degree as d_i, the complexity is

$$\sum_{j=2,\cdots,m_0} O(d_j \cdot (j-1) \cdot \log m_0) = O(m_0|E_0| \log m_0) \tag{9}$$

where E_0 is the set of edges connected to source nodes A_0.

With the *GreedyShaving* procedure, our scalable algorithm can be designed so as to generate candidate suspicious source node sets. In our implementation, we use singular vector decomposition (SVD) for our algorithm. Each top singular vector gives a spectral view of high connectivity communities. However, those singular vectors are not associated with suspiciousness scores. Thus combined with the top singular vectors, our fast greedy algorithm is given in Alg. 3.

Algorithm 3 *FastGreedy* Algorithm for Fraud detection.

Given bipartite multigraph $\mathcal{G}(U, V, E)$.
$\mathbb{L}$ = get first several left singular vectors
for all $L^{(k)} \in \mathbb{L}$ **do**
  Rank source nodes U decreasingly on $L^{(k)}$
  $\tilde{U}^{(k)}$ = truncate $u \in U$ when $L_u^{(k)} \leq \frac{1}{\sqrt{|U|}}$
  $GreedyShaving$ with initial $\tilde{U}^{(k)}$.
end for
return the best A^* with maximized objective $HS(A^*)$,
  and the rank of $v \in V$ by $f_{A^*}(v) \cdot P(v|A^*)$.

Theorem 3.3 (Algorithm complexity). *In the graph $\mathcal{G}(U,V,E)$, given $|V| = O(|U|)$ and $|E| = O(|U|^{\epsilon_0})$, the complexity of FastGreedy algorithm is subquadratic, i.e., $o(|U|^2)$ in little-o notation, if the size of truncated user set $|\tilde{U}^{(k)}| \leq |U|^{1/\epsilon}$, where $\epsilon > \max\{1.5, \frac{2}{3-\epsilon_0}\}$.*

Proof. The *FastGreedy* algorithm executes *GreedyShaving* in a constant iterations. A_0 is assigned to $\tilde{U}^{(k)}$ in every *GreedyShaving* procedure. Then $m_0 = |A_0| = |\tilde{U}^{(k)}|$. In the adjacency matrix $\mathbb{M}$ of the graph, we consider the submatrix $\mathbb{M}_0$ with A_0 as rows and V as columns. If the fraudulent dense block is in submatrix $\mathbb{M}_0$, then we assume that the block has at most $O(m_0)$ columns. Excluding the dense block, the remaining part of $\mathbb{M}_0$ is assumed to have the same density with the whole matrix $\mathbb{M}$. Therefore, the total number of edges in $\mathbb{M}_0$ is

$$O(|E_0|) = O(m_0^2 + \frac{m_0 \cdot |E|}{|U|}) = O(|U|^{2/\epsilon} + |U|^{1/\epsilon - 1}|E|)$$

Then based on equation (9), the algorithm complexity is

$$O(m_0|E_0| \log m_0) = O((|U|^{3/\epsilon} + |U|^{2/\epsilon - 1 + \epsilon_0}) \log |U|)$$

Therefore, if $\epsilon > \max\{1.5, \frac{2}{3-\epsilon_0}\}$, then the complexity is subquadratic $o(|U|^2)$. ∎

In real life graph, $\epsilon_0 \leq 1.6$, so if $\epsilon > 1.5$ the complexity of *FastGreedy* algorithm is subquadratic. Therefore, without loss of performance and efficiency, we can limit $|\tilde{U}^{(k)}| \leq |U|^{1/1.6}$ for truncating an ordered U in the *FastGreedy* algorithm for a large dataset.

In *FastGreedy* algorithm for HS-α, SVD on adjacency matrix $\mathbb{M}$ is used to generate initial blocks for the *GreedyShaving* procedure. Although we can still use SVD on $\mathbb{M}$ for HS with holistic attributes, yet considering attributes of timestamps and rating scores may bring more benefits. Observing that not every combination of # of stars, timestamps and product ids has a value in a multi-way tenor representation, we can only choose every existing triplets (*object*, *timestamp*, #*stars*) as one column, and *user* as rows, to form a new matrix. The above transformation is called the *matricization* of a tensor, which outputs a new matrix. With proper time bins, e.g., one hour or day, and re-clustering of #*stars*, the flattening matrix becomes more dense and contains more attribute information. Thus we use such a flattening matrix with each column weighted by the sudden-drop suspiciousness for our *FastGreedy* algorithm.

4 EXPERIMENTS

In the experiments, we only consider the significant multiple bursts for fluctuated time series of sink nodes. We keep those awakening-burst point pairs with the altitude difference Δc at least 50% of the largest altitude difference in the time series. Table 2 gives the statistics of our six datasets which are publicly available for academic research [3]. Our extensive experiments showed that the performance was insensitive to scaling base b, and became very stable when larger than 32. Hence we choose $b = 32$ in the following experiments.

4.1 Evaluation on different injection density

In the experiments, we mimic the fraudsters' behaviors and randomly choose 200 objects with indegree no more than 100 as the fraudsters' customers, since less popular objects are more susceptible to manipulation. On the other hand, the fraudulent accounts can come from the hijacked user accounts. Thus we uniformly sample out a number of users as fraudsters from the whole user set. To test on different fraudulent density, the number of sampled fraudsters

[3]Yelp dataset is from https://www.yelp.com/dataset_challenge

Table 2: Data Statistics

Data Name	#nodes	#edges	time span
BeerAdvocate [25]	26.5K x 50.8K	1.07M	Jan 08 - Nov 11
Yelp	686K x 85.3K	2.68M	Oct 04 - Jul 16
Amazon Phone & Acc [24]	2.26M x 329K	3.45M	Jan 07 - Jul 14
Amazon Electronics [24]	4.20M x 476K	7.82M	Dec 98 - Jul 14
Amazon Grocery [24]	763K x 165K	1.29M	Jan 07 - Jul 14
Amazon mix category [26]	1.08M x 726K	2.72M	Jan 04 - Jun 06

ranges from 200 to 20,000. Those fraudsters as a whole randomly create 200 fake edges to each of the 200 products. As a results, the fraudulent density ranges from 1.0 to 0.01 for testing. The rating time is generated for each fraudulent edge: first randomly choose a start time between the earliest and the latest creation time of the existing edges; and then plus a randomly and biased time interval sampled from intervals of the exiting creation time, to mimic the surge of fraudsters' attacks. Besides, a high rating score, e.g. 4 or 4.5, is randomly chosen for each fraudulent edge[4]. At the same time, we also add camouflage to other product nodes with the same number of fraudulent edges.

Fig. 1a shows the results of HS-α on the BeerAdvocate data, compared with Fraudar and SpokEn that only consider topology information as HS-α does. To detect fraudsters of a low density is much harder than that of a high density, so the better methods are able to detect fraudsters of lower density with a high accuracy. Since HS-α only considers the topology information in our novel contrast suspiciousness, we compare HS-α with the baselines based on graph topology. When the fraudulent density decreases from the right to the left along the horizontal axis, HS-α can detect fraudulent density as low as 0.125 in a high F measure, better than 0.8 which is the best of the baselines.

Fig. 1b shows the results of HoloScope HS, which uses topology, temporal and rating attributes. Comparing to those baselines on the same kinds of attributes, HS can keep as high F measure as more than 90% before reaching 0.033 in density, better by far than the baseline methods (0.50 in density). In other words, to create the same amount of fraudulent edges, HS can detect fraud with high accuracy even when fraudsters use 6,000 source nodes (user accounts). On the other hand, the best of the baselines detects in a low accuracy (less than 50%) even when only 600 source nodes are used, which is easier to detect. Besides, HS using several signals further improves over HS-α with only topology signal (compared with Fig. 1a) by decreasing density from 0.125 to 0.033 with high detection accuracy.

In order to give a comparison on all six data sets with different injection density, we propose to use the two metrics: a low-case "auc" and the lowest *detection density*, described in the notes of Table 3. The table reports the fraud detection results of our HoloScope (HS) and the baselines on the six datasets. Since the accuracy curve stops at 0.01 (the minimum testing density); and we add zero accuracy at zero density, the ideal value of auc is 0.995. The auc on source and sink nodes are reported separately. In terms of sink nodes, HS outputs the rank list by suspiciousness scores, we use the area under the upper-case AUC (similar to F-measure) accuracy curve

[4] the injection code is also open-sourced for reproducibility

Table 3: Experimental results on real data with injected labels

Data Name	metrics*	source nodes				sink nodes			
		M-Zoom	D-Cube	CrossSpot	HS	M-Zoom	D-Cube	CrossSpot	HS
BeerAdvocate	auc	0.7280	0.7353	0.2259	**0.9758**	0.6221	0.6454	0.1295	**0.9945**
	F≥90%	0.5000	0.5000	–	**0.0333**	0.5000	0.5000	–	**0.0333**
Yelp	auc	0.9019	0.9137	0.9916	**0.9925**	0.9709	0.8863	0.0415	**0.9950**
	F≥90%	0.2500	0.2000	0.0200	**0.0143**	0.0250	1.0000	–	**0.0100**
Amazon Phone & Acc	auc	0.9246	0.8042	0.0169	**0.9691**	0.9279	0.8810	0.0515	**0.9823**
	F≥90%	0.1667	0.5000	–	**0.0200**†	0.1429	0.1000	–	**0.0200**†
Amazon Electronics	auc	0.9141	0.9117	0.0009	**0.9250**	0.9142	0.7868	0.0301	**0.9385**
	F≥90%	0.2000	0.1250	–	**0.1000**	**0.1000**	0.5000	–	0.1250
Amazon Grocery	auc	0.8998	0.8428	0.0058	**0.9250**	0.8756	0.8241	0.0200	**0.9621**
	F≥90%	0.1667	0.5000	–	**0.1000**	0.1250	0.2500	–	**0.1000**
Amazon mix category	auc	0.9001	0.8490	0.5747	**0.9922**	0.9937	0.9346	0.0157	**0.9950**
	F≥90%	0.2500	0.5000	0.2000†	**0.0167**	**0.0100**	0.2000	–	**0.0100**

* we use the two metrics: the area under the curve (abbrev as low-case "auc") of the accuracy curve as drawn in Fig. 1b, and the lowest *detection density* that the method can detect in high accuracy (F measure ≥ 90%).
† one of the above fraudulent density was not detected in high accuracy.

along all testing density. As the table suggests, our HS achieved the best auc among the baselines, and even reached the ideal auc in two cases.

Furthermore, we compare the lowest *detection density* in Table 3. The better a method is, the lower density it should be able to detect well. As we can see, HS has the smallest detection density in most cases, which can be as small as $^{200}/_{14000}$= 0.0143 on source nodes, and reached the minimum testing density of 0.01 on sink nodes. That means we can detect fraudsters in high accuracy even when they use 14 thousand accounts to create fake edges for each of 200 objects, due to the holistic use of signals in contrast suspiciousness framework. The fraudulent objects can also be detected accurately.

4.2 Evaluation on Sina Weibo with real labels

We also did experiments on a large real dataset from Sina Weibo, which has 2.75 million users, 8.08 million messages, and 50.1 million edges in Dec 2013. The user names and ids, and message ids are from the online system. Thus we can check their existence status in the system to evaluate the experiments. If the messages or the users were deleted from the system, we treat them as the basis for identifying suspicious users and messages. Since it is impossible to check all of the users and messages, we firstly collected a candidate set, which is the union of the output sets from the HS and the baseline methods. The real labels are from the candidate set by checking the status whether they still exists in Sina Weibo (checked in Feb. 2017). We used a program on the API service of Sina Weibo to check the candidate user and message id lists, finally resulting in 3957 labeled users and 1615 labeled messages.

The experimental results in Fig. 1c show that HS achieved high F-measure on accuracy, which detected 3781 labeled users higher than M-Zoom's 1963 labeled users. The F-measure of HS improved about 30% and 60%, compared with M-Zoom and D-Cube respectively. CrossSpot biased to include a large amount of users ($> 500,000$) in their detection results, which recalled less than 150 extra labeled users, getting very low F-measure, less than 1.5%. In terms of messages, the HS achieved around 0.8 in AUC from the ranking list of the results, while M-Zoom and D-Cube got lower recall, and CrossSpot still suffered from very low F-measure with many false positive messages. Therefore, our HoloScope outperformed the baselines in real-labeled data as well.

4.3 Scalability

To verify the complexity, we choose two representative datasets: BeerAdvocate data which has the highest volume density, and Amazon Electronics which has the most edges. We truncated the two datasets according to different time ranges, i.e., from the past 3 months, 6 months, or several years to the last day, so that the generated data size increases. Our algorithm is implemented in Python. As shown in Fig. 2, the running time of our algorithm increases almost linearly with the number of the edges.

5 CONCLUSION

In conclusion, we proposed a fraud detection method, HoloScope, on a bipartite graph which can have timestamps and rating scores. HoloScope has the following advantages: 1) **Novel suspiciousness metric:** we propose a dynamic *contrast suspiciousness* metric, which emphasizes the contrast in behavior between fraudsters and honest users in terms of topology, temporal spikes, and rating deviation. Including the temporal and rating information, HS performs even better than HS-α that only considers topology. Moreover, the unified suspiciousness framework can integrate more signals easily in a natural way, e.g. the divergence of topic distributions between text reviews from fraudsters and honest users. 2) **Robustness and theoretical analysis of fraudsters' obstruction:** we showed that if the fraudsters use less than a lower bound of time to rate an object, they will cause a suspicious drop or burst. The robustness is guaranteed that no matter how the fraudsters manipulate the rating pace in a less time cost, they will be caught suspicious. In other words, our HoloScope can obstruct fraudsters and increases their time cost. 3) **Effectiveness:** we achieved higher accuracy on both semi-real and real datasets than the baselines under the circumstance of camouflage. 4) **Scalability:** under reasonable assumptions, the algorithm is sub-quadratic in the number of nodes.

6 ACKNOWLEDGEMENTS

This material is based upon work supported by National Grand Fundamental Research 973 Program of China under Grant No. 2014CB340401, the Beijing National Science Foundation under Grant No. 4172059, the National Science Foundation of China under Grant No. 61772498, and the National Science Foundation under Grant No. IIS-1247489, CNS-1314632, IIS-1408924. Shenghua Liu is also supported by the scholarship from China Scholarship Council.

Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation, or other funding parties.

REFERENCES

[1] Leman Akoglu, Rishi Chandy, and Christos Faloutsos. 2013. Opinion Fraud Detection in Online Reviews by Network Effects. *ICWSM* 13 (2013), 2–11.

[2] Leman Akoglu, Mary McGlohon, and Christos Faloutsos. 2010. Oddball: Spotting anomalies in weighted graphs. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 410–421.

[3] Miguel Araujo, Stephan Günnemann, Gonzalo Mateos, and Christos Faloutsos. 2014. Beyond blocks: Hyperbolic community detection. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 50–65.

[4] Alex Beutel, Wanhong Xu, Venkatesan Guruswami, Christopher Palow, and Christos Faloutsos. 2013. Copycatch: stopping group attacks by spotting lockstep behavior in social networks. In *Proceedings of the 22nd international conference on World Wide Web*. ACM, 119–130.

[5] Moses Charikar. 2000. Greedy approximation algorithms for finding dense components in a graph. In *International Workshop on Approximation Algorithms for Combinatorial Optimization*. Springer, 84–95.

[6] Gordon V Cormack, Charles LA Clarke, and Stefan Buettcher. 2009. Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*. ACM, 758–759.

[7] Geli Fei, Arjun Mukherjee, Bing Liu, Meichun Hsu, Malu Castellanos, and Riddhiman Ghosh. 2013. Exploiting Burstiness in Reviews for Review Spammer Detection. *ICWSM* 13 (2013), 175–184.

[8] David Freedman and Persi Diaconis. 1981. On the histogram as a density estimator: L 2 theory. *Probability theory and related fields* 57, 4 (1981), 453–476.

[9] Maria Giatsoglou, Despoina Chatzakou, Neil Shah, Christos Faloutsos, and Athena Vakali. 2015. Retweeting activity on Twitter: signs of deception. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 122–134.

[10] David Gibson, Ravi Kumar, and Andrew Tomkins. 2005. Discovering large dense subgraphs in massive graphs. In *Proceedings of the 31st international conference on Very large data bases*. VLDB Endowment, 721–732.

[11] Nikou Günnemann, Stephan Günnemann, and Christos Faloutsos. 2014. Robust multivariate autoregression for anomaly detection in dynamic product ratings. In *Proceedings of the 23rd international conference on World wide web*. ACM, 361–372.

[12] Stephan Günnemann, Nikou Günnemann, and Christos Faloutsos. 2014. Detecting anomalies in dynamic rating data: A robust probabilistic model for rating evolution. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 841–850.

[13] Bryan Hooi, Hyun Ah Song, Alex Beutel, Neil Shah, Kijung Shin, and Christos Faloutsos. 2016. Fraudar: bounding graph fraud in the face of camouflage. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 895–904.

[14] Meng Jiang, Alex Beutel, Peng Cui, Bryan Hooi, Shiqiang Yang, and Christos Faloutsos. 2015. A general suspiciousness metric for dense blocks in multimodal data. In *Data Mining (ICDM), 2015 IEEE International Conference on*. IEEE, 781–786.

[15] Meng Jiang, Alex Beutel, Peng Cui, Bryan Hooi, Shiqiang Yang, and Christos Faloutsos. 2016. Spotting Suspicious Behaviors in Multimodal Data: A General Metric and Algorithms. *IEEE Transactions on Knowledge and Data Engineering* 28, 8 (2016), 2187–2200.

[16] Meng Jiang, Peng Cui, Alex Beutel, Christos Faloutsos, and Shiqiang Yang. Catching synchronized behaviors in large networks: a graph mining approach. *Knowledge Discovery from Data, ACM Transactions on* 10, 4 (????), 35:1–35:27.

[17] Meng Jiang, Peng Cui, Alex Beutel, Christos Faloutsos, and Shiqiang Yang. 2014. CatchSync: catching synchronized behavior in large directed graphs. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 941–950.

[18] Meng Jiang, Peng Cui, Alex Beutel, Christos Faloutsos, and Shiqiang Yang. 2014. Inferring strange behavior from connectivity pattern in social networks. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 126–138.

[19] Qing Ke, Emilio Ferrara, Filippo Radicchi, and Alessandro Flammini. 2015. Defining and identifying Sleeping Beauties in science. *Proceedings of the National Academy of Sciences* 112, 24 (2015), 7426–7431.

[20] Efstathios Kirkos, Charalambos Spathis, and Yannis Manolopoulos. 2007. Data mining techniques for the detection of fraudulent financial statements. *Expert systems with applications* 32, 4 (2007), 995–1003.

[21] Ravi Kumar, Jasmine Novak, Prabhakar Raghavan, and Andrew Tomkins. 2005. On the bursty evolution of blogspace. *World Wide Web* 8, 2 (2005), 159–178.

[22] Huayi Li, Geli Fei, Shuai Wang, Bing Liu, Weixiang Shao, Arjun Mukherjee, and Jidong Shao. 2016. Modeling Review Spam Using Temporal Patterns and Co-bursting Behaviors. *arXiv preprint arXiv:1611.06625* (2016).

[23] Pankaj Malhotra, Lovekesh Vig, Gautam Shroff, and Puneet Agarwal. 2015. Long short term memory networks for anomaly detection in time series. In *Proceedings*. Presses universitaires de Louvain, 89.

[24] Julian McAuley and Jure Leskovec. 2013. Hidden factors and hidden topics: understanding rating dimensions with review text. In *Proceedings of the 7th ACM conference on Recommender systems*. ACM, 165–172.

[25] Julian John McAuley and Jure Leskovec. 2013. From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews. In *Proceedings of the 22nd international conference on World Wide Web*. ACM, 897–908.

[26] Arjun Mukherjee, Bing Liu, and Natalie Glance. 2012. Spotting fake reviewer groups in consumer reviews. In *Proceedings of the 21st international conference on World Wide Web*. ACM, 191–200.

[27] Seth A Myers and Jure Leskovec. 2014. The bursty dynamics of the twitter information network. In *Proceedings of the 23rd international conference on World wide web*. ACM, 913–924.

[28] White Ops. 2016. The Methbot Operation. http://go.whiteops.com/rs/179-SQE-823/images/WO_Methbot_Operation_WP.pdf. (2016). [Online; accessed Jan-7-2017].

[29] Shashank Pandit, Duen Horng Chau, Samuel Wang, and Christos Faloutsos. 2007. Netprobe: a fast and scalable system for fraud detection in online auction networks. In *Proceedings of the 16th international conference on World Wide Web*. ACM, 201–210.

[30] Ashwin Paranjape, Austin R Benson, and Jure Leskovec. 2016. Motifs in Temporal Networks. *arXiv preprint arXiv:1612.09259* (2016).

[31] B Aditya Prakash, Ashwin Sridharan, Mukund Seshadri, Sridhar Machiraju, and Christos Faloutsos. 2010. Eigenspokes: Surprising patterns and scalable community chipping in large graphs. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 435–448.

[32] Stephen Robertson. 2004. Understanding inverse document frequency: on theoretical arguments for IDF. *Journal of documentation* 60, 5 (2004), 503–520.

[33] Neil Shah, Alex Beutel, Bryan Hooi, Leman Akoglu, Stephan Gunnemann, Disha Makhija, Mohit Kumar, and Christos Faloutsos. 2015. EdgeCentric: Anomaly Detection in Edge-Attributed Networks. *arXiv preprint arXiv:1510.05544* (2015).

[34] Kijung Shin, Tina Eliassi-Rad, and Christos Faloutsos. 2016. CoreScope: Graph Mining Using k-Core Analysis-Patterns, Anomalies and Algorithms. ICDM.

[35] Kijung Shin, Bryan Hooi, and Christos Faloutsos. 2016. M-Zoom: Fast Dense-Block Detection in Tensors with Quality Guarantees. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. 264–280.

[36] Kijung Shin, Bryan Hooi, Jisu Kim, and Christos Faloutsos. 2017. D-Cube: Dense-Block Detection in Terabyte-Scale Tensors. In *Proceedings of the 10th ACM International Conference on Web Search and Data Mining (WSDM '17)*. ACM.

[37] Karen Sparck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation* 28, 1 (1972), 11–21.

[38] Herbert A Sturges. 1926. The choice of a class interval. *J. Amer. Statist. Assoc.* 21, 153 (1926), 65–66.

[39] Jun-ichi Takeuchi and Kenji Yamanishi. 2006. A unifying framework for detecting outliers and change points from time series. *IEEE transactions on Knowledge and Data Engineering* 18, 4 (2006), 482–492.

[40] Sihong Xie, Guan Wang, Shuyang Lin, and Philip S Yu. 2012. Review spam detection via temporal pattern discovery. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 823–831.

[41] Jaewon Yang and Jure Leskovec. 2011. Patterns of temporal variation in online media. In *Proceedings of the fourth ACM international conference on Web search and data mining*. ACM, 177–186.