Locating Data Flow Anomaly With Statistical Distance

Ruoyu Wang

***Abstract*—[abstract]**

I. Introduction

Big data industry has been blooming within this decade, reshaping the form of life and work across the globe. According to [17], 1.2ZB data had been produced from 2012 to 2014. And the amount doubles every two years. Currently, there are totally 2.7ZB data in the digital universe [1]. Big data analysis has been widely adopted in scientific experiments [20], electric business [2, 29, 9], healthcare [11], governments [16] and many other fields.

However, it will be harder in the future to harness the exploding volumes of data since problems have already appeared in data management and engineering, threatening trustworthiness and reliability of data flows inside working systems. Data error rate in enterprises is approximately 1% to 5%, and for some, even above 30% [23].

Those data anomalies may arise due to both internal and external reasons with respect to a certain system. From one hand, components inside the system may generate problematic source data. For example, in a sensor network, some sensors may generate erroneous data when it experiences power failure or other extreme conditions [22]. Data packages will be lost if sensor nodes fail to connect to network or some sensor hub goes down [12]. Also, human operators act as a heavily vulnerable part to bugs and mistakes. Some even deliberately modify system configurations for malicious compromises [24]. A study [31] found that 65% of organizations state that human errors are the main cause of data problems.

On the other hand, data manipulation [14] from outside hackers composes another potential threat of data quality and reliability. Taken Apache Hadoop as an example. It's security issues has long been discussed within communities and industry [26, 30, 15, 27]. As is shown in Figure 1, the two basic vulnerabilities: *lack of access control* and the *absence of encryption* expose the entire cluster to dangerous threats. Data flows can be intercepted and modified; services can also be altered and blocked [13]. Although there are several frameworks(e.g. Kerberos, Sentry, Knox etc.) and algorithms providing basic protection [39, 28, 33, 37, 10], clever attackers can always bypass the barriers and sneak into the core of data pipelines. Several approaches are developed as sentinels to detect probable infiltrations. However, these approaches are not able to locate corrupted data under carefully planned manipulations. Nor can they figure out the exact reasons and recover the original records. To locate and diagnose anomalies

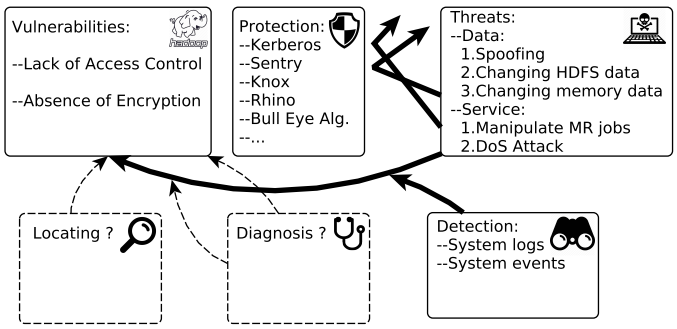

Fig. 1. Security Issues of Hadoop Clusters

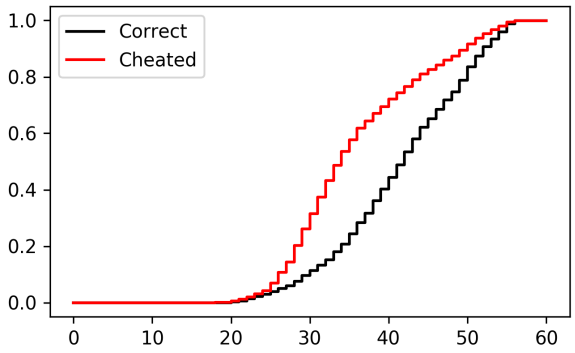

Fig. 2. Example Cumulative Distribution Function of Original And Modified Data

in data pipelines under carefully planned and disguised data manipulations are still on demand by industry and academia.

According to our observation, typical data manipulations on numerical data will lead to the drift of its distribution. For example, a Taobao online seller's transaction records can be "click farmed" to increase the volume of sales. Fig. 2 shows the sales distribution within one day. The curve for cheated data is emulated according to a popular method for click farming. It can be observed that there exists clearly a gap between these two distributions. These anomalies inside data pipelines will severely affect mining and learning algorithms and further change the final decision given by the entire system.

In order to address issues resulted from data manipulation and pipeline errors, we propose a novel mechanism to detect and locate corrupted data within a data pipeline via statistical

distance. As far as we know, this algorithm is the first attempt against data anomaly via statistical distance. Evaluations are performed on synthetic and real-world data sets, demonstrating the correctness and effectiveness of the mechanism.

The rest of the paper is organised as follow: Section II introduces preliminaries and recent works on data anomaly detection and statistical distance. Algorithm details are proposed in section III. Section IV presents evaluation results and further findings of the algorithm. And all contents are concluded in section V.

II. Preliminaries and Related Work

A. Data Anomaly Detection

Anomaly detection, also known as outlier detection, has been studied for a long time and discussed in diverse research domains, such as fraud detection, intrusion detection, system monitoring, fault detection and event detection in sensor networks. According to a systematic classification in [8], anomaly detection algorithms deal with input data in the form of points(or records), sequences, graphs and spatial relationships, where point data is the simplest and well studied, others are attracting more attention in new studies.

Prevalent anomalies can be classified into *Point Anomalies*, *Contextual(or Conditional) Anomalies* and *Collective Anomalies*. Point anomalies refers to an individual data instance that is considered anomalous with respect to others. But if it is anomalous only in certain circumstances or a specific context, the instance is regarded as contextual anomaly. If a group of related data(e.g. a segment of sequence) instances is anomalous with respect to other groups in the data set(e.g. the entire sequence), it is called a collective anomaly.

Detection approaches can be categorized into three types according to whether data is labeled: *Supervised*, *Semi-Supervised* and *Unsupervised* anomaly detection. As the name suggests, supervised detection methods train models on completely labeled data while unsupervised detection leverages data without any labeling. Semi-supervised detection approaches train model on data that has labeled instances for only the normal class. Supervised detection is commonly applied when both normal and anomalous data can be obtained. When it comes to the circumstances that anomalous data is hard to obtain or there exist too many diverse types of anomalies to enumerate, semi-supervised or unsupervised approaches are usually taken into consideration.

To make the final decision, detection algorithms mostly yield a score from each input instance, denoting how likely it is anomalous. The algorithm then selects top few as anomalies or compare the score with a threshold. Or, detection algorithms output a label on each instance, then decide whether each label belongs to the normal class.

Currently, distance based [5, 4] and feature evolving algorithms [19, 18, 25] algorithms seize most attention. Others adopted tree isolation [38], model based [35] and statistical methods [40] in certain applications.

To detect collective anomalies, [6] adopted the *ART(Adoptive Resonance Theory)* neural networks to detect time-series anomalies. *Box Modeling* is proposed in [7]. And *Longest Common Subsequence* was leveraged in [3] as similarity metric for symbolic sequence. Markovian modeling techniques are also popular in this domain[34, 32, 21]. [36] depicted groups in social media as combinations of different "roles" and compare groups according to the proportion of each role within each group.

B. Click Farming Detection

C. Statistical Divergence

Statistical divergence, also called statistical distance, is a function which describes the "distance" of one probability distribution to the other on a statistical manifold. Suppose S is a space of probability distributions, then a divergence is a function from S to non-negative real numbers:

$$D(\cdot||\cdot) : S \times S \to \mathbb{R}^+ \tag{1}$$

Divergence between two distributions P and Q, written as $D(P||Q)$, satisfies:

1) $D(P||Q) \geq 0, \forall P, Q \in S$
2) $D(P||Q) = 0$, if and only if $P = Q$

There are many ways to calculate divergence, such as f-divergences, M-divergences and S-divergences. Some of them provides better properties which brings conveniences to the design and implementation of our approach.

1) Kullback-Leibler Divergence: P, Q are discrete probability distributions, $Q(i) = 0$ implies $P(i) = 0$ for $\forall i$, the *Kullback-Leibler Divergence* from Q to P is defined to be:

$$KLD(P||Q) = \sum_{Q(i)\neq 0} P(i) log\Big(\frac{P(i)}{Q(i)}\Big) \tag{2}$$

2) Jensen-Shannon Divergence: P, Q are discrete probability distributions, *Jensen-Shannon Divergence* between P and Q is defined to be:

$$JSD(P||Q) = \frac{1}{2}KLD(P||M) + \frac{1}{2}KLD(Q||M) \tag{3}$$

where $M = \frac{1}{2}(P + Q)$.

A more generalized form is defined to be:

$$JSD_{\pi_1,\ldots,\pi_n}(P_1, \ldots, P_n) = \sum_{i=1}^{n} \frac{1}{\pi_i} KLD(P_i||M) \tag{4}$$

where $M = \sum_{i=1}^{n} \frac{1}{\pi_i} P_i$ and $\sum_{i=1}^{n} \frac{1}{\pi_i} = 1$.

Jensen-Shannon divergence has some fine properties:

1) $JSD(P||Q) = JSD(Q||P), \forall P, Q \in S$.
2) $0 \leq JSD_{\pi_1,\ldots,\pi_n}(P_1, \ldots, P_n) \leq log_k(n)$. If a k based algorithm is adopted.
3) To calculate $JSD(P||Q)$, it need not necessarily to be true that $Q(i) = 0$ implies $P(i) = 0$.

[If P, Q are continuous distribution?]

III. Algorithm Details

Diverse data sets in the real world show certain structures which may be resulted from hidden patterns or relationships among records in a collection of data. For example, the volume of vehicles in the highway and the business transaction records, they may show a relatively stable distribution in the daily scale. Manipulation on those data(e.g. Fig 2) results in a drift or distortion of the distribution, which can be captured to trigger the alarm. Although the population parameters(e.g. mean, variance, etc.) are unknown and usually impossible to obtain, it can be sampled and estimated according to the central limitation theorem.

A. Technical Points

[Can be divided and settled inside later two subsections]

1) Which classifier should be chosen?
2) How to determine the classifier threshold?**[fixed value, 3σ]**
3) How to locate the compromised component?
4) How to deal with slightly drifting distribution?
5) .[to be continued ...]

B. Divergence-Based Collective Anomaly Detection

Suppose data chunks in the given data set S are groups of instances sampled from a population driven by a static distribution. And we are given in advance an evidence set E which contains $n(n \geq 2)$ collections of correct sample data. Then each data collection in S can be checked by the following algorithm.

Algorithm 1 Basic Classification

Input: Evidence set $E = \{D_1, \ldots, D_n\}$; New data collection D'
Output: Whether D' is anomalous

1: **for** $i \leftarrow 1$ to n do **do**
2: $P_i \leftarrow$ the distribution of D_i
3: **end for**
4: $M \leftarrow \frac{1}{n}\sum_{i=1}^{n} P_i$
5: **for** $i \leftarrow 1$ to n do **do**
6: $J_i \leftarrow JSD(P_i||M)$
7: **end for**
8: $N \leftarrow$ normal distribution estimated from $J_1, \ldots, J_n$
9: $P' \leftarrow$ distribution of D'
10: $J' \leftarrow JSD(P'||M)$
11: $p \leftarrow$ probability density of J' in N
12: **if** $p <$ threshold T **then**
13: **Return** True
14: **else**
15: **Return** False
16: **end if**

As shown in Algorithm 1, n evidence collections are used to estimate the ground truth population distribution M. Then n evidence divergences are calculated, composing a gaussian classifier to classify the new distribution sample. Although it is convenient to compute $JSD(P_1, \ldots, P_n, P')$ instead of $JSD(P_1||M), \ldots, JSD(P_n||M), JSD(P'||M)$. It is not suitable for classification. Jensen-Shannon divergence of $n+1$ distributions will dilute the affection of the abnormal one, in which case the difference between P' being normal and anomalous will become subtle when n goes larger.

Similar to the fact that sampling values around a certain parameter will yield a gaussian distribution, sampling divergences around a certain population distribution yields a gaussian distribution $N(\mu, \sigma)$ where μ is a value slightly larger than zero. μ can not be zero according to line 4. M takes into consideration all existing values in every distribution sample and averages corresponding probabilities. Thus M may consist entries that does not exist in P_i and probability in certain entries in M may vary from that in P_i. For example, suppose $P_1(1) = 0.5, P_1(2) = 0.3, P_1(3) = 0.2$; $P_2(1) = 0.3, P_2(2) = 0.4, P_2(3) = 0.3$; $P_3(1) = 0.3, P_3(3) = 0.5, P_3(4) = 0.2$, then $M(1) = \frac{11}{3}, M(2) = \frac{7}{3}, M(3) = \frac{10}{3}, M(4) = \frac{2}{3}$. None of P_1, P_2, P_3 is the same with M. Although the distance cannot be negative values, normal distribution is the closest to the distribution of all JSD values.

C. Distribution Histogram

Not all data collections can be assumed subject to known distribution models. Parameter approximation will probably lead to errors above our expectation. For example, the distribution of vehicle volume in the highway may not correspond to any of the prevalent distribution model. Few assumption of parameters can be made in advance on such data. For the sake of generality, non-parametric estimation should be applied.

Surely, the kernel density estimation approaches will give a smooth, continuous distribution curve under on any sampled data. The computational cost will be much higher than applying a coarse histogram approximation. Moreover, even if we applied a distribution model upon a collection of sampled data, yielding more accurate results, the computation of divergence will be much more complex on continuous functions than discrete distributions.

Histogram methods can not only be applied to non-parametric distributions, but also to miscellaneous data types. From a general point of view, as long as there exits a function mapping the input data instance into a subset of real numbers, this data collection can be counted into a histogram. For example, date, time and single characters.

In order to generate a accurate approximation, step size is the most important parameter the algorithm should determine. If the size is too small then the resulting histogram will be over fitting; but if the size is too large then the estimation will be too coarse to depict the original shape. According to statistics theory, when dealing with a sample size of k, a step size of

$$l = c\sigma k^{-0.2} \tag{5}$$

will give a best partition, where c is a constant relative to the shape of distribution(e.g. for normal distribution, $c = 1.05$).

D. Threshold

One important factor in the algorithm is the value of threshold. A higher threshold rejects more instances, improving the sensitivity of anomalous data while increasing the number of false alarms. A lower threshold provide higher true negative rates yet neglecting more possible threats.

A naive but prevalent approach is to set a fixed value as threshold. This approach is easy to implement and may give satisfying results in specific cases. However, a fixed threshold requires analysis of the specific problem, manual observation and tuning of parameters, which involves lots of human labor. The rule of "3σ" can be used to automatically determine a threshold. But in some cases that the anomalous data intersecting at the boundary with normal instances, the threshold may fail more often to discover those manipulated ones.

However, applying divergence as the distance metric among data collection distributions provides a fine property. That is, divergences of normal data collections assemble together, forming a quasi-normal distribution. And those of anomalous data collections lies in the right-hand-side interval on the real number axis. As a result of further observation, the divergence of anomalous data collections also assemble to be a quasi-normal distribution, since statistical distance has been restricted by a strict upper bound.

[Better give a point drawing for each JSD value]

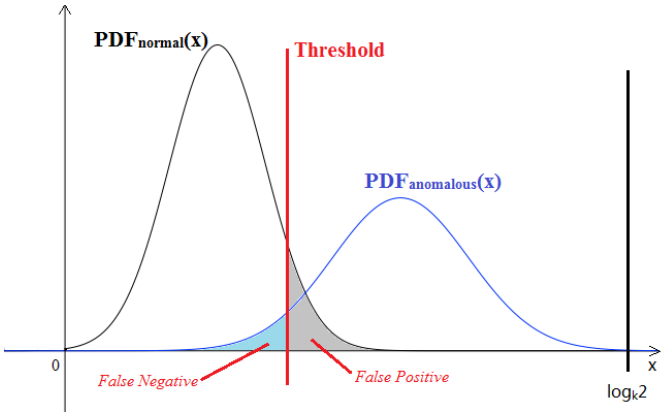

Fig. 3. Threshold Example

As shown in Fig 3, the black curve($PDF_{normal}(x)$) displays the probability density function(PDF) fitting those JSDs calculated from normal data collections; the blue curve($PDF_{anomalous}(x)$) displays the PDF derived from anomalous data collections. Threshold is to minimize total error(False Negative and False Positive).

Suppose:

$$PDF_{normal}(x) \approx N(\mu_n, \sigma_n) \tag{6}$$
$$PDF_{anomalous}(x) \approx N(\mu_a, \sigma_a) \tag{7}$$

Then the optimal threshold T is:

$$\begin{aligned} T &= \arg\min_T \int_0^T PDF_{anomalous}(x)dx + \int_T^{log_k 2} PDF_{normal}(x)dx \\ &\approx \arg\min_T \int_{-\infty}^{T} \frac{e^{-\frac{(x-\mu_a)^2}{2\sigma_a^2}}}{\sqrt{2\pi}\sigma_a}dx + \int_T^{+\infty} \frac{e^{-\frac{(x-\mu_n)^2}{2\sigma_n^2}}}{\sqrt{2\pi}\sigma_n}dx \\ &= \frac{\mu_n\sigma_a + \mu_a\sigma_n}{\sigma_a + \sigma_n} \end{aligned} \tag{8}$$

According to equation(8), threshold value can be automatically determined. The optimal threshold will minimize total errors, yielding an optimal outcome. The only requirement for this technique is another evidence set containing data point derived from anomalous data collections.

However, this is not accurate enough, since equation(8) implicate an assumption that the chances are the same for a new data collection to be either anomalous or not. If we can determine the probability for a new data collection to be anomalous in any segment of data sequence, the equation should be modified as:

$$\begin{aligned} T &= \arg\min_T \alpha \int_0^T PDF_{anomalous}(x)dx + (1-\alpha)\int_T^{log_k 2} PDF_{normal}(x)dx \\ &\approx \arg\min_T \alpha \int_{-\infty}^{T} \frac{e^{-\frac{(x-\mu_a)^2}{2\sigma_a^2}}}{\sqrt{2\pi}\sigma_a}dx + (1-\alpha)\int_T^{+\infty} \frac{e^{-\frac{(x-\mu_n)^2}{2\sigma_n^2}}}{\sqrt{2\pi}\sigma_n}dx \end{aligned} \tag{9}$$

Where α is the anomaly probability. Equation(9) is much more complicated than (8). It can be reduced to a quadratic equation. But may not have real roots in all cases.

From another point of view, equation (8) can be regarded as a weighted averaging between two mean value: μ_n and μ_a. From this aspect, we can derive some similar approximations:

$$T = \frac{\mu_n\sqrt{\sigma_a} + \mu_a\sqrt{\sigma_n}}{\sqrt{\sigma_a} + \sqrt{\sigma_n}} \tag{10}$$
$$T = \frac{\mu_n\sigma_a\sqrt{-ln(1-\alpha)} + \mu_a\sigma_n\sqrt{-ln\alpha}}{\sigma_a\sqrt{-ln(1-\alpha)} + \sigma_n\sqrt{-ln\alpha}} \tag{11}$$
$$T = \frac{\mu_n\sqrt{|ln[\sigma_a(1-\alpha)]|} + \mu_a\sqrt{|ln(\sigma_n\alpha)|}}{\sqrt{|ln[\sigma_a(1-\alpha)]|} + \sqrt{|ln(\sigma_n\alpha)|}} \tag{12}$$

Due to the restriction sample capacity, the replacement of JSD distributions by guassian distribution may lose too much accuracy. Compared with equation (10), (11)and (12), (9) may not give a best result.

E. Dynamic DCAD

Another important factor that should be seriously considered is the assumption in Algorithm 1. In most cases, real world data is always in the process of evolution and fluctuation. Therefore, the algorithms should automatically adapt to the trend at any time. To meet the need, a sliding window technique has been applied to the basic algorithm frame work. Considering discussion in section III-C and III-D, a dynamic version is proposed as follow:

Algorithm 2 Dynamic DAD

Input: Normal evidence set $E_N = \{D_{N_1}, \ldots, D_{N_n}\}$ by given order; Anomalous evidence set $E_A = \{D_{A_1}, \ldots, D_{A_m}\}$ by given order; New data collection D'; Estimated anomalous probability α

Output: Whether D' is anomalous

1: **for** $i \leftarrow 1$ to n **do**
2: $P_{N_i} \leftarrow$ histogram of D_{N_i}
3: **end for**
4: **for** $i \leftarrow 1$ to m **do**
5: $P_{A_i} \leftarrow$ histogram of D_{A_i}
6: **end for**
7: $P' \leftarrow$ histogram of D'
8: $M \leftarrow \frac{1}{n}\sum_{i=1}^{n} P_{N_i}$
9: **for** $i \leftarrow 1$ to n **do**
10: $J_{N_i} \leftarrow JSD(P_{N_i}||M)$
11: **end for**
12: **for** $i \leftarrow 1$ to m **do**
13: $J_{A_i} \leftarrow JSD(P_{A_i}||M)$
14: **end for**
15: $J' = JSD(P'|||M)$
16: $(\mu_N, \sigma_N) \leftarrow$ estimated from $\{J_{N_1}, \ldots, J_{N_n}\}$
17: $(\mu_A, \sigma_A) \leftarrow$ estimated from $\{J_{A_1}, \ldots, J_{A_m}\}$
18: $T \leftarrow$ proper threshold derived from (μ_N, σ_N), (μ_A, σ_A) and α
19: **if** $J' < T$ **then**
20: $E_N \leftarrow E_N \backslash \{D_{N_0}\} \cup \{D'\}$
21: **Return** False
22: **else**
23: $E_A \leftarrow E_A \backslash \{D_{A_0}\} \cup \{D'\}$
24: **Return** True
25: **end if**

Algorithm 2 uses E_N and E_A as two sliding windows, keeping up with latest trend of both normal and anomalous features. Thus, the two window sizes n and m cannot be too large. Otherwise the evolving features will be flattened and aligned to the older ones. For the sake of estimation accuracy, they cannot be too small either. The size should refer to ratio of both data updates and feature evolution.

Line 20 and 23 update the evidence sets, replacing the most outdated one, moving the sliding window forward one step ahead. Moreover, α can also be updated according to a history sequence of return values, if necessary.

IV. Evaluation

A. Experiment Environment

The algorithm was implemented and interpreted in Python 3.6. All experiments was tested on Ubuntu 17.04.

B. Methodology

Raise and answer some research questions. Present test background and methods.

We adopted a data set containing Taobao online sellers' transaction records[1] provided by Alibaba Tian Chi big data competition. The data package contains seller features data set, user payments data set and user browsing behaviour data set. An example of the user payments data set is shown in Table I.

TABLE I
User Payment Record Exmaple

User ID	Seller ID	Payment Time
10523185	1629	2015-11-11 17:00:00
2	2	2
3	3	3
⋮	⋮	⋮

We randomly chose one seller(ID: 1629) and extracted transaction history of this seller, records ranging from Nov. 11th 2015 to Oct. 31st 2016. Entire transaction set was then divided into 325 collections, each containing records in one day.

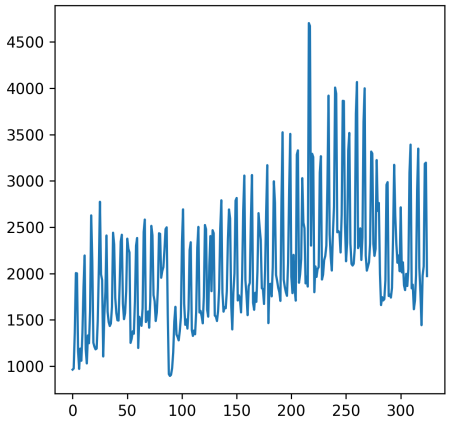

Fig. 4. Daily Transaction Volumes

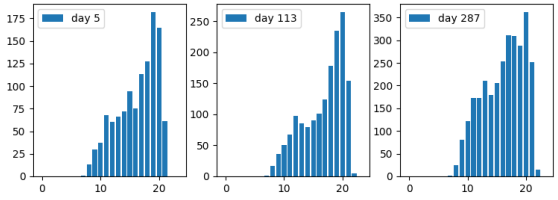

Fig. 5. Randomly selected 3 days and draw histograms by hourly sale

Fig. 4 shows the daily transaction volume of the seller. It sees a trend of daily sale which is slightly moving as the time goes. And the daily selling distributions, as is shown in Fig. 5, display a roughly similar shape.

Click farmed data was generated according to patterns described in online reports**[ref here]**. The most popular way of click farming is to call online a group of people, assign each of them an amount of transaction. And those "employees" individually buy certain products via different buyer accounts.

[1] **https://tianchi.aliyun.com/competition/information.htm?raceId=231591**.

Therefore, a significant feature of this approach is that the cheating transactions usually assemble together in a short period of time. It is called "Centralized Farming". Some other "employers" are more clever. They usually arrange a time table for each employee to create new transactions. Thus the transaction distribution may not vary too much with and without click farming. This is called "Equalized Farming".

To emulate "Centralized Click Farming", we randomly inserted some gaussian-distributed transactions in a chosen collection, where the total number of new records is the same of the original volume. To emulate a extremely clever "Equalized Click Farmer", we simply doubled each record in the chosen collection to make the new distribution exactly the same as the original one, which is harder for the online platform to discover.

To play the role of purchasing platform, we surveillance two levels of transaction distribution. The first level is simply drawing a histogram on time spans. The second level is to draw a histogram on the sub-volumes of each time span. For example, shown in Fig. 6, the first level histogram can be drawn by counting the number of hourly sales. The frequencies of each bucket in 1st level histogram is: $F = \{0, 0, 0, 0, 0, 0, 0, 1, 13, 30, 37, 68, 60, 66, 72, 94, 75, 113, 127, 182, 165, 61, 0, 0\}$. Then, counting F with a step size 20 gives the 2nd level histogram.

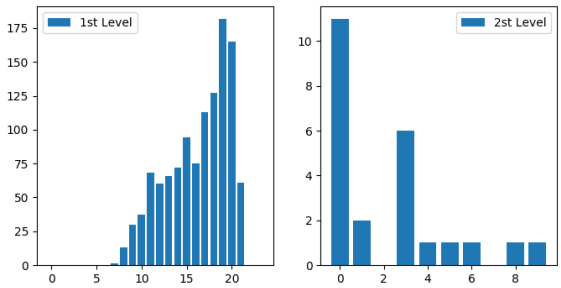

Fig. 6. Example of 1st and 2nd Level Histogram

In order to test the performance of Algorithm 2, every single collection was fed into the algorithm by the time order. Each collection input has a probability of α to be anomalous, namely, click farmed. The algorithm checks each collection with 1st or 2nd level histogram and give an answer of true and false. Then a F1 score was computed as the result of the experiment.

C. Experiment on Raw Data

We first tested Algorithm 2 on raw data set in order to see whether and why the algorithm works. Due to desensitization process, time stamps in raw data set only tell which hour the transaction was committed. Thus, the histogram was drawn by counting hourly volume of each collection. For $\alpha = 0.2$, we select first 30 days as the evidence set of normal collections; day 21st to 30th were also emulated to be click farmed, composing the evidence set of anomalous collections. Equation(11) was employed as the threshold calculator. The results are shown in Table II, where *true positive rate*, *false positive rate* and *accuracy* were recorded.

TABLE II
CLASSIFICATION RESULTS ON RAW DATA

	Centralized			Equalized		
	TPR	FAR	ACC	TPR	FAR	ACC
1st Level	100	4.05	96.72	26.64	28.03	62.49
2nd Level	88.01	7.68	91.41	83.89	15.78	84.07

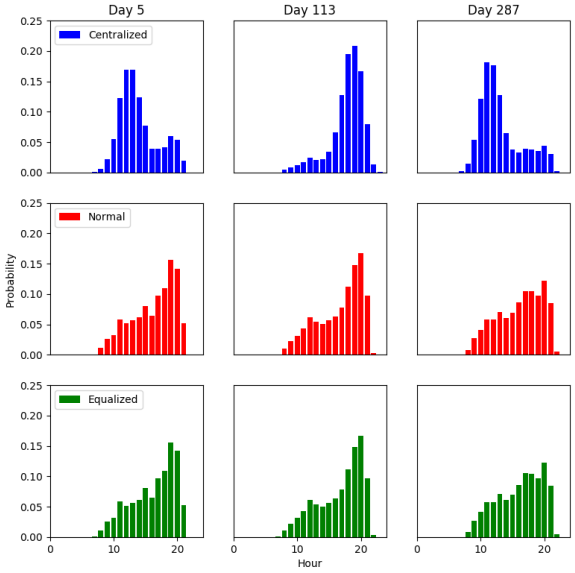

Fig. 7. 1st Level Histogram of Raw Data

When classifying toward 1st level histograms, centralized click farming behaviours can be easily discovered. As displayed in the left two columns in Fig. 7, normal collections share a similar distribution while centralized click farmed ones abruptly violates the original shape. However, as a clever click farmer, equalized click farming did not in the least distorted the distribution. Most of them escaped the security check under the perfect disguise. But when it comes to 2nd level histogram, the "clever disguise" does not work any longer. It can be clearly seen in Fig. 8 that both types of click farming shows an obvious difference from the normal ones, while the normal ones still share a similar distribution. From the overview in Fig. 9 and 10, the difference between normal and anomalous collections can be observed more intuitively.

D. Experiment on Synthetic Data

As is put in the former section, every single record in the transaction set only gives a time stamp aligned at hours. Part of the information was erased with the absence of minutes and seconds. In the following experiments, Every time stamp was assigned a random value for minutes and seconds.

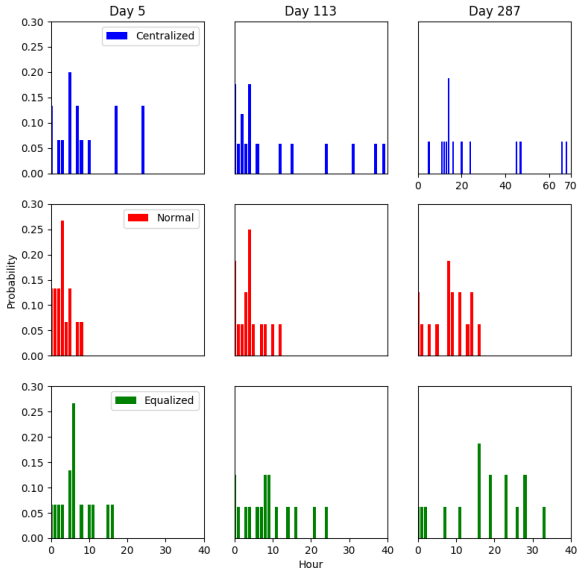

Fig. 8. 2nd Level Histogram of Raw Data

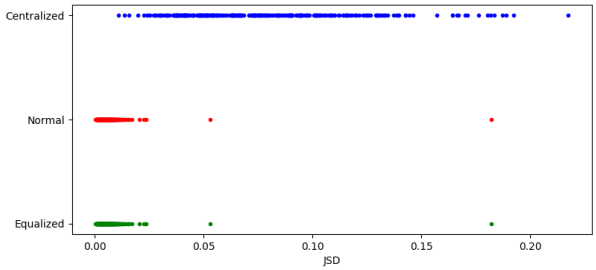

Fig. 9. JSD of 1st Level Histograms

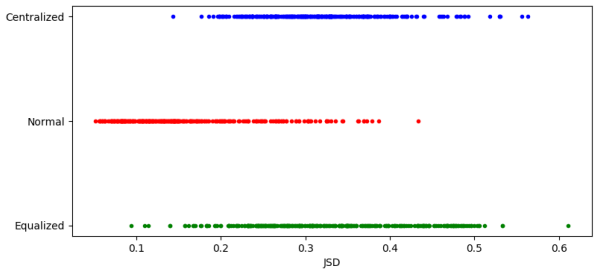

Fig. 10. JSD of 2nd Level Histogram

Therefore, the synthetic data set should be closer to the real world case. Moreover, the step size in the histograms can now be calculated more precisely. In the 1st level histogram, the constant c in equation(5) was set to 0.5 since the PDF of daily distribution was approximately a linear function. And in the 2nd level histogram, sub-volumes of every 300 seconds were counted, producing more data points in the 2nd level data collections. Then, constant c in equation(5) was also set to 0.5 given that the 2nd level histogram was also approximately a linear function. We tested algorithm performance for $\alpha \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$, with E_N and E_A the same configuration as before. Results were shown in Table III and IV.

TABLE III
1ST LEVEL

	Centralized			Equalized		
α	TPR(%)	FPR(%)	ACC(%)	TPR(%)	FPR(%)	ACC(%)
0.1	86.10	1.14	97.40	5.75	12.28	79.66
0.2	96.43	9.83	91.30	26.97	26.47	64.29
0.3	97.95	16.49	87.68	37.46	39.46	53.67
0.4	99.18	31.94	81.30	57.23	48.32	53.79
0.5	99.33	23.40	88.25	67.67	58.23	54.46
0.6	99.62	25.70	89.83	67.03	73.78	51.64
0.7	100.00	32.51	90.51	69.69	76.28	56.16
0.8	99.72	26.97	94.35	75.56	69.44	67.00
0.9	99.63	15.73	98.19	76.59	75.18	70.73

TABLE IV
2ND LEVEL

	Centralized			Equalized		
α	TPR(%)	FPR(%)	ACC(%)	TPR(%)	FPR(%)	ACC(%)
0.1	75.60	3.90	93.74	97.84	1.64	98..31
0.2	91.93	18.43	93.73	99.42	7.39	94.01
0.3	95.01	23.70	81.69	100.00	23.93	83.05
0.4	98.09	28.66	82.49	100.00	24.59	84.75
0.5	98.33	36.54	80.00	100.00	23.76	88.36
0.6	98.35	40.69	83.41	100.00	35.94	84.63
0.7	98.54	37.97	87.80	100.00	31.27	90.84
0.8	98.47	40.38	91.19	100.00	33.04	92.66
0.9	98.51	47.56	94.58	100.00	28.44	97.40

The result shows that the algorithm can efficiently distinguish the normal and anomalous data collections. No matter how much proportion of data is anomalous, the algorithm always gives a high accuracy. That is because the adaptive threshold was designed to minimize the number of total errors, which was irrelevant to TPR or FPR. When α was increasing, there were less data collections added into E_N. Thus the update of E_N became slower and FPR went higher. On the contrary, TPR became higher as α increased. Moreover, since $|E_N| > |E_A|$, it needs more collection instances for $|E_N|$ then that for $|E_A|$ to keep up to the trend. Therefore, it was faster of FPR to increase than TPR to increase. As a result of that, there were more false alarms emerged and ACC went lower, as α increased from 0.1 to 0.5. After that, however, although FPR was still increasing, total number of normal instances decreased dramatically and those false alarms contributed less to total errors. Therefore, ACC rose up again.

Fig. 11 shows the ROC curve of the gaussian classifier dealing with centralized click farming with 1st level histogram and equalized click farming with 2nd level histogram.**[Comment: This shows that the classifier is efficient]**

[Next, show a figure of ROC curve for different click farm magnitude. Explore how sensitive the classifier is for the click farming problem. This requires lots of experiment data. Not finished yet]

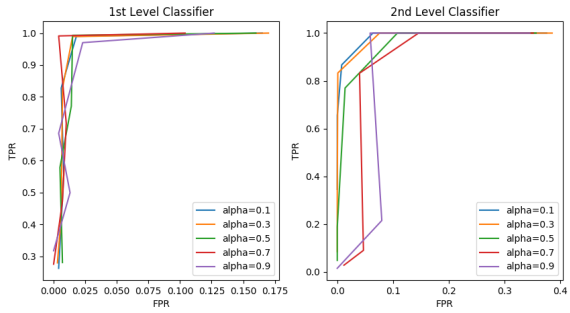

Fig. 11. ROC Curve of 1st and 2nd Level Classifier

This is the figure

Fig. 12. ROC Curve of Different Magnitude of Click Farming

E. Threshold Optimization

give results of the comparison of different threshold equations, in table V

[Comment: Which equation is the best in each case? Why?]

F. Comparison with other collective classification algorithms

It would be better to have this comparison, but it requires much more experiments and the data set may not be very suitable.

V. Conclusion

Acknowledgement

References

[1] *Big Data Statistics And Facts for 2017*. 2017. URL: https://www.waterfordtechnologies.com/big-data-interesting-facts/.

[2] Nathan Bronson, Thomas Lento, and Janet L Wiener. "Open data challenges at facebook". In: *Data Engineering (ICDE), 2015 IEEE 31st International Conference on*. IEEE. 2015, pp. 1516–1519.

[3] Suratna Budalakoti et al. "Anomaly detection in large sets of high-dimensional symbol sequences". In: (2006).

[4] Lei Cao et al. "Multi-Tactic Distance-Based Outlier Detection". In: *Data Engineering (ICDE), 2017 IEEE 33rd International Conference on*. IEEE. 2017, pp. 959–970.

[5] Lei Cao et al. "Scalable distance-based outlier detection over high-volume data streams". In: *Data Engineering (ICDE), 2014 IEEE 30th International Conference on*. IEEE. 2014, pp. 76–87.

[6] T Caudell and D Newman. "An adaptive resonance architecture to define normality and detect novelties in time series and databases". In: *IEEE World Congress on Neural Networks, Portland, Oregon*. 1993, pp. 166–176.

[7] Philip K Chan and Matthew V Mahoney. "Modeling multiple time series for anomaly detection". In: *Data Mining, Fifth IEEE International Conference on*. IEEE. 2005, 8–pp.

[8] Varun Chandola, Arindam Banerjee, and Vipin Kumar. "Anomaly detection: A survey". In: *ACM computing surveys (CSUR)* 41.3 (2009), p. 15.

[9] Guoqiang Jerry Chen et al. "Realtime data processing at facebook". In: *Proceedings of the 2016 International Conference on Management of Data*. ACM. 2016, pp. 1087–1098.

[10] Jason C Cohen and Subrata Acharya. "Towards a trusted HDFS storage platform: Mitigating threats to Hadoop infrastructures using hardware-accelerated encryption with TPM-rooted key protection". In: *Journal of Information Security and Applications* 19.3 (2014), pp. 224–244.

[11] Peter Groves et al. "The'big data'revolution in healthcare: Accelerating value and innovation". In: (2016).

[12] Herodotos Herodotou et al. "Scalable near real-time failure localization of data center networks". In: *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2014, pp. 1689–1698.

[13] Jingwei Huang, David M Nicol, and Roy H Campbell. "Denial-of-service threat to Hadoop/YARN clusters with multi-tenancy". In: *Big Data (BigData Congress), 2014 IEEE International Congress on*. IEEE. 2014, pp. 48–55.

[14] *Is Data Manipulation the Next Step in Cyber Crime*. URL: https://www.cloudmask.com/blog/is-data-manipulation-the-next-step-in-cybercrime.

[15] Masoumeh Rezaei Jam et al. "A survey on security of Hadoop". In: *Computer and Knowledge Engineering (ICCKE), 2014 4th International eConference on*. IEEE. 2014, pp. 716–721.

[16] Gang-Hoon Kim, Silvana Trimi, and Ji-Hyong Chung. "Big-data applications in the government sector". In: *Communications of the ACM* 57.3 (2014), pp. 78–85.

[17] Emmanuel Letouzé and Johannes Jütting. "Official statistics, big data and human development: towards a new conceptual and operational approach". In: *Data Pop Alliance and PARIS21* (2014).

[18] Yaliang Li et al. "On the discovery of evolving truth". In: *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM. 2015, pp. 675–684.

[19] Mohammad M Masud et al. "Classification and adaptive novel class detection of feature-evolving data streams". In: *IEEE Transactions on Knowledge and Data Engineering* 25.7 (2013), pp. 1484–1497.

[20] Frank Austin Nothaft et al. "Rethinking data-intensive science using scalable analytics systems". In: *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*. ACM. 2015, pp. 631–646.

[21] Dmitry Pavlov. "Sequence modeling with mixtures of conditional maximum entropy distributions". In: *Data Mining, 2003. ICDM 2003. Third IEEE International Conference on*. IEEE. 2003, pp. 251–258.

[22] Murad A Rassam, Mohd Aizaini Maarof, and Anazida Zainal. "Adaptive and online data anomaly detection for

TABLE V
COMPARE OF DIFFERENT THRESHOLD CALCULATOR

Equation	Synthetic	Gaussian
$T = \frac{\sigma_B \mu_A + \sigma_A \mu_B}{\sigma_B + \sigma_A}$	1.0588	0.5810
$T = \frac{\sqrt{\sigma_B}\mu_A + \sqrt{\sigma_A}\mu_B}{\sqrt{\sigma_B} + \sqrt{\sigma_A}}$	2.1861	0.5718
$T = \frac{(ln\sigma_B)\mu_A + (ln\sigma_A)\mu_B}{ln\sigma_B + ln\sigma_A}$	2.2390	0.3860
$T = \frac{(\sqrt{\lvert ln\sigma_B \rvert})\mu_A + (\sqrt{\lvert ln\sigma_A \rvert})\mu_B}{\sqrt{\lvert ln\sigma_B \rvert} + \sqrt{\lvert ln\sigma_A \rvert}}$	2.2082	0.5515
$T = \frac{(ln\sqrt{\sigma_B})\mu_A + (ln\sqrt{\sigma_A})\mu_B}{ln\sqrt{\sigma_B} + ln\sqrt{\sigma_A}}$	2.1211	0.3811
$T = \frac{\sigma_B(1-\alpha)\mu_A + \sigma_A\alpha\mu_B}{\sigma_B(1-\alpha) + \sigma_A\alpha}$	0.7688	0.3515
$T = \frac{(\sigma_B\sqrt{1-\alpha})\mu_A + (\sigma_A\sqrt{\alpha})\mu_B}{\sigma_B\sqrt{1-\alpha} + \sigma_A\sqrt{\alpha}}$	0.8158	0.4510
$T = \frac{\left[\sigma_B ln(1-\alpha)\right]\mu_A + \left[\sigma_A ln\alpha\right]\mu_B}{\sigma_B ln(1-\alpha) + \sigma_A ln\alpha}$	1.5227	0.5523
$T = \frac{\mu_A\sigma_B\sqrt{-ln(1-\alpha)} + \mu_B\sigma_A\sqrt{-ln\alpha}}{\sigma_B\sqrt{-ln(1-\alpha)} + \sigma_A\sqrt{-ln\alpha}}$	2.6013	0.5791
$T = \frac{\mu_A\sigma_B ln\sqrt{1-\alpha} + \mu_B\sigma_A ln\sqrt{\alpha}}{\sigma_B ln\sqrt{1-\alpha} + \sigma_A ln\sqrt{\alpha}}$	1.4155	0.5578
$T = \frac{\mu_A\sqrt{\lvert ln\left[\sigma_B(1-\alpha)\right] \rvert} + \mu_B\sqrt{\lvert ln(\sigma_A\alpha) \rvert}}{\sqrt{\lvert ln\left[\sigma_B(1-\alpha)\right] \rvert} + \sqrt{\lvert ln(\sigma_A\alpha) \rvert}}$	2.2730	0.5600
$T = \frac{\mu_A ln\sqrt{\sigma_B(1-\alpha)} + \mu_B ln\sqrt{\sigma_A\alpha}}{ln\sqrt{\sigma_B(1-\alpha)} + ln\sqrt{\sigma_A\alpha}}$	2.0753	0.2808
$T = \arg\min_{\mu_A \le T \le \mu_B} \alpha \cdot \int_{-\infty}^{T} P_B(x)dx + (1-\alpha) \cdot \int_{T}^{+\infty} P_A(x)dx$	1.9300	0.6072

wireless sensor systems". In: *Knowledge-Based Systems* 60 (2014), pp. 44–57.

[23] Barna Saha and Divesh Srivastava. "Data quality: The other face of big data". In: *Data Engineering (ICDE), 2014 IEEE 30th International Conference on*. IEEE. 2014, pp. 1294–1297.

[24] Felix Schuster et al. "VC3: Trustworthy data analytics in the cloud using SGX". In: *Security and Privacy (SP), 2015 IEEE Symposium on*. IEEE. 2015, pp. 38–54.

[25] Junming Shao, Zahra Ahmadi, and Stefan Kramer. "Prototype-based learning on concept-drifting data streams". In: *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2014, pp. 412–421.

[26] Ather Sharif et al. "Current security threats and prevention measures relating to cloud services, Hadoop concurrent processing, and big data". In: *Big Data (Big Data), 2015 IEEE International Conference on*. IEEE. 2015, pp. 1865–1870.

[27] Priya P Sharma and Chandrakant P Navdeti. "Securing big data hadoop: a review of security issues, threats and solution". In: *Int. J. Comput. Sci. Inf. Technol* 5.2 (2014), pp. 2126–2131.

[28] Biplab Sikdar. "Spatio-Temporal Correlations in Cyber-Physical Systems: A Defense Against Data Availability Attacks". In: *Proceedings of the 3rd ACM Workshop on Cyber-Physical System Security*. ACM. 2017, pp. 103–110.

[29] Roshan Sumbaly, Jay Kreps, and Sam Shah. "The big data ecosystem at linkedin". In: *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*. ACM. 2013, pp. 1125–1134.

[30] Duygu Sinanc Terzi, Ramazan Terzi, and Seref Sagiroglu. "A survey on security and privacy issues in big data". In: *Internet Technology and Secured Transactions (ICITST), 2015 10th International Conference for*. IEEE. 2015, pp. 202–207.

[31] TowerData. *4 Steps to Eliminating Human Error in Big Data*. 2013. URL: http://www.towerdata.com/blog/bid/113787/4-Steps-to-Eliminating-Human-Error-in-Big-Data (visited on 06/30/2017).

[32] Christina Warrender, Stephanie Forrest, and Barak Pearlmutter. "Detecting intrusions using system calls: Alternative data models". In: *Security and Privacy, 1999. Proceedings of the 1999 IEEE Symposium on*. IEEE. 1999, pp. 133–145.

[33] Zhang Xu et al. "High fidelity data reduction for big data security dependency analyses". In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM. 2016, pp. 504–516.

[34] Nong Ye et al. "A markov chain model of temporal behavior for anomaly detection". In: *Proceedings of the 2000 IEEE Systems, Man, and Cybernetics Information Assurance and Security Workshop*. Vol. 166. West Point, NY. 2000, p. 169.

[35] Jianhua Yin and Jianyong Wang. "A model-based approach for text clustering with outlier detection". In: *Data Engineering (ICDE), 2016 IEEE 32nd International Conference on*. IEEE. 2016, pp. 625–636.

[36] Rose Yu, Xinran He, and Yan Liu. "Glad: group anomaly detection in social media analysis". In: *ACM Transactions on Knowledge Discovery from Data (TKDD)* 10.2 (2015), p. 18.

[37] Xianqing Yu, Peng Ning, and Mladen A Vouk. "Enhancing security of Hadoop in a public cloud". In: *Information and Communication Systems (ICICS), 2015 6th International Conference on*. IEEE. 2015, pp. 38–43.

[38] Xuyun Zhang et al. "LSHiForest: A Generic Framework for Fast Tree Isolation Based Ensemble Anomaly Anal-

ysis". In: *Data Engineering (ICDE), 2017 IEEE 33rd International Conference on*. IEEE. 2017, pp. 983–994.

[39] Zhiyuan Zheng and AL Reddy. "Towards Improving Data Validity of Cyber-Physical Systems through Path Redundancy". In: *Proceedings of the 3rd ACM Workshop on Cyber-Physical System Security*. ACM. 2017, pp. 91–102.

[40] Yunyue Zhu and Dennis Shasha. "Statstream: Statistical monitoring of thousands of data streams in real time". In: *Proceedings of the 28th international conference on Very Large Data Bases*. VLDB Endowment. 2002, pp. 358–369.