Dynamic Anomaly Detection for Collective Data via Statistical Distance

Ruoyu Wang [#1], Daniel Sun [*2], Guoqiang Li [#3]

[#] *School of Software, Shanghai Jiao Tong University*
Shanghai, 200240, China
[1] trams.wang.ruoyu@gmail.com
[3] li.g@sjtu.edu.cn

[*] *Data61, Commonwealth Scientific and Industrial Research Organization, Australia*
[2] daniel.sun@data61.csiro.au

***Abstract*—Hackers can infiltrate networks and find their way into databases and software systems. Not only can they steal chunks of data and hold them for ransom, but also alter key information contained in those systems, setting up traps hard to discover and causing damages unlikely to recover. Although altering single record may not oblige correct value scope such that it is hard to detect, the change, especially to a subset of data, can be observed and detected in a collective scale. In this paper, we present a collective anomaly detection technique based on statistical distance. The technique extracts distribution similarities among data collections, and then uses the statistical distance to detect collective anomalies. In order to adapt to data dynamics, our technique continuously evaluates metrics as evolving features. Aiming at broad generalisation, this technique can replace several components of our solution to further improve the performance of detection for various scenarios. To illustrate details of the technique and explore its efficiency, we case-studied a real world problem of click farming detection against malicious online sellers. The evaluation shows that the technique maps normal and anomalous data collections into two distinct clusters by yielding efficient classifiers. Those classifiers are sensitive sufficiently to a much smaller magnitude of data alerting, compared with real world malicious behaviours–that is–it is applicable in the real world.**

I. Introduction

Major improvements in data acquisition, transmission and storage are driving increased data availability and affordability. Currently, there are totally 2.7ZB data in the digital universe [1] and the growing speed is doubling every two years. It has already been and will be much harder in the future to harness the exploding volumes of data which is now pervasive leading to many problems in data management and engineering, threatening trustworthiness and reliability of data flows inside working systems. In 2016, according to the research by Tricentis [2], software failures cost $1.1 trillion US dollars in total and those occurred at 363 investigated companies affected 4.4 billion customers, causing the total lost time of more than 315.5 years. And it takes more than 45% of total expense to eliminate those errors [3]. System configuration problems, as a matter of fact, are also responsible for the disturbing situation. They take even dominant places in failures in large distributed clusters [4].

Data error rate in enterprises is approximately 1% to 5%, and for some, even above 30% [5]. Those data anomalies may arise due to both internal and external reasons with respect to a certain system. On 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 [6]. Data packages will be lost if sensor nodes fail to connect to network or some sensor hubs break down [7]. Also, human operators act as a heavily vulnerable part to bugs and mistakes. Malicious insiders even deliberately modify system configurations for fatal compromises [8]. A study [9] found that 65% of organizations state that human errors are the main cause of data problems.

On the other hand, data manipulation [10] from outside hackers composes another potential threat of data quality and reliability. *Data Manipulation* here, according to a NSA definition, refers to that "hackers can infiltrate networks via any attack vector, find their way into databases and applications and change information contained in those systems, rather than stealing data and holding it for ransom". If data inside the system is compromised, they will severely affect mining and learning algorithms and further change the final decision given by the entire system. In 2013, hackers from Syria put up fake reports via Associated Press' Twitter account and caused a 150-point drop in the Dow [11].

However, from the aspect of data itself, we can still detect malicious data manipulation behaviours. According to our observation, typical data manipulations on numerical data will lead to a drift or distortion of its original distribution. With the reshaping measurable, we can enclose data collections with similar distribution patterns inside the same cluster and filter out those strange shaped ones. In this paper, we adopt Jensen-Shannon divergence as the similarity measurement among different distributions. Comparing with a estimated ground truth, each data collection can be mapped into a single real number within a definite interval. Then a gaussian classifier can be applied to detect outliers derived from manipulated data. To automatically calculate adaptive threshold for the classifier, we keep two evidence sets for both normal points and anomalies, taking advantage of the property provided by Jensen-Shannon divergence. And those evidence sets are set to be slide windows to keep up to the evolving features in

dynamic scenarios. Details of the technique are put together with a real world problem of "Click Farming" and had been evaluated on both raw and synthetic data sets corresponding to the problem, demonstrating the correctness and effectiveness of the technique.

The rest of the paper is organised as follows: Section II states related works on data anomaly detection and describes a real world problem. Section III introduces statistical distance. Details of the technique are proposed in section IV. Then section V presents evaluation results and further findings of the algorithm. Finally, all contents are concluded and discussed in section VI.

II. 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 comprehensive study in [12], 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 anomaly 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 instances(e.g. a segment of sequence) is anomalous with respect to other groups in the data set(e.g. the entire sequence), it is called a collective anomaly [13].

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. [14] When it comes to the circumstances that anomalous data is hard to obtain or there exist too many 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 [15], [16] and feature evolving algorithms [17], [18], [19] algorithms seize most attention. Others adopted tree isolation [20], model based [21] and statistical methods [22] in certain applications.

To detect collective anomalies, [23] adopted the *ART(Adoptive Resonance Theory)* neural networks to detect time-series anomalies. *Box Modeling* is proposed in [24]. And *Longest Common Subsequence* was leveraged in [25] as similarity metric for symbolic sequence. Markovian modeling techniques are also popular in this domain[26], [27], [28]. [29] depicted groups in social media as combinations of different "roles" and compare groups according to the proportion of each role within each group.

B. Real World Problem: Click Farming Detection

Since the establishment of Taobao in 2003, its online business platform has been growing from a follower in the market to the No.1 service provider in B2B and B2C markets, possessing a market share of 50.6% to 56.2% in China by 2016 [30]. Currently, there are more than 9.4 million sellers in Taobao, providing more than 1 billion different products. Under the super-pressure caused by massive competitors, most of the sellers choose to use some cheating techniques to raise reputation and product sale volumes, then improve rankings in search lists.

The most popular approach to manipulate transaction and reputation data is *Click Farming*, where sellers use a large number of dedicated customer accounts to create fake transaction records and give high remarks on products. Professional click farmers are usually well organized groups or companies containing thousands of people. Leaders of such groups receive orders from sellers and assign tasks to other members to create fake transactions. Some companies even develop professional applications that can be deployed on common PCs to improve productivity [31].

A research performed in China showed that 81.9% of the people investigated had heard of the behaviour of click farming, 51.2% of them had seen people around them click farmed and 18.9% of them had experience of click farming themselves [32]. American researchers reported in 2015 that over 11000 sellers were detected to have click farmed records and only 2.2% of 4000 investigaed sellers had been penalized because of the cheating attempts [33].

Current detection techniques for click farming mainly focus on user behaviours, such as browsing frequencies and periods, most common purchasing time, favourite products, remarks and whether they communicate with sellers [34]. Those techniques require the platform to keep lots of records and user features. However, the detection can be easily bypassed by trained workers and some well programmed applications.

Although it is hard to classify users as honest and malicious, we can still find clues from the sellers' aspect. For normal sellers, their customers are usually similar since choices of products are seldom changed. Therefore, the distribution of transactions in a fixed period of time, say one day, is relatively stable. No matter how much alike between honest users and robots or the employed workers, the fake transaction records will always cause a bias or distortion of the original transaction distribution. To better observe the problem, we downloaded a

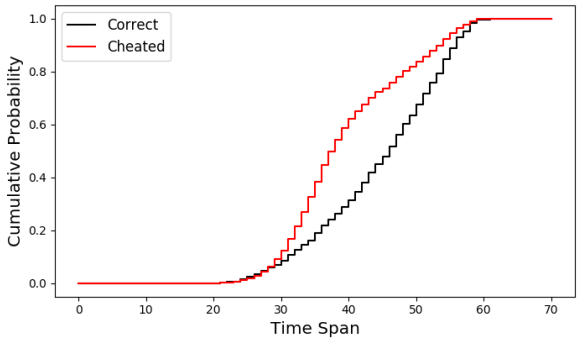

Fig. 1. Example Cumulative Distribution Function of Original And Click Farmed Daily Transaction Data

real world data set containing Taobao online sellers' transaction records and emulated the circumstances if it had been click farmed(see section V-A). Fig. 1 shows the difference between normal and click farmed distributions of one day in the data set. Thus, if we can measure the similarity between different transaction distributions, there is still a chance for us to detect dishonest sellers.

III. Preliminaries

Statistical divergence, also called statistical distance, measures the similarity between two or more distributions. Mathematically, statistical divergence is a function which describes the "distance" of one probability distribution to the other on a statistical manifold. Let $\mathbb{S}$ be a space of probability distributions, then a divergence is a function from $\mathbb{S}$ to non-negative real numbers:

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

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

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

For our purposes, we do not require the function D to have the property: $D(P||Q) = D(Q||P)$. But we do need it to be true that if Q is more similar with P than U, then $D(Q||P) < D(U||P)$. There are many ways to calculate divergence, such as f-divergences, M-divergences and S-divergences. Considering that the sensitivity and computational cost of the function are directly related to the performance of our technique, we chose Jensen-Shannon divergence, which is derived from Kullback-Leibler divergence, as the measurement.

A. Kullback-Leibler Divergence

Let P, Q be 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:

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

For P, Q being continuous distributions:

$$KLD(P||Q) = \int_{-\infty}^{+\infty} P(x) log \frac{P(x)}{Q(x)} dx \tag{3}$$

B. Jensen-Shannon Divergence

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

$$JSD(P||Q) = \frac{1}{2} KLD(P||M) + \frac{1}{2} KLD(Q||M) \tag{4}$$

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

A more generalized form is defined to be:

$$JSD_{\pi_1,\ldots,\pi_n}(P_1, \ldots, P_n) = \sum_{i=1}^{n} \frac{1}{\pi_i} KLD(P_i||M) \tag{5}$$

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) $JSD(P||Q) = JSD(Q||P), \forall P, Q \in \mathbb{S}$.
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 $P(i) = 0$.

IV. Statistical Detection

Diverse data sets in the real world show certain structures caused by hidden patterns or relationships among records in a collection of data. For example, the traffic volume 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. 1) results in a drift or distortion of the distribution, which can be captured to trigger the alarm. Main idea of the solution is simple. But there are several difficulties to be solved in the specific design and implementation of the technique:

1) Are the corresponding one-dimensional points distinguishable after mapping form normal and anomalous distributions? And which classifier is efficient towards those points?
2) Is there a general approach that can effectively depict different distributions in various circumstances?
3) What classifier specific metric should be adopted to filter out anomalies? Can it be further optimized?
4) Online shops are often in the process of expanding or dwindling. Thus the sales distribution will slightly change as time goes on. How to make the technique adapt to these drifting distributions?

These questions are addressed in the following subsections correspondingly. IV-A answers the first question and introduces a basic framework of our technique. The second question is studied in IV-B. Then IV-C gives an analysis about adaptive metric that minimizes total error. And a dynamic framework is proposed in IV-D.

Algorithm 1 Basic Classification

Input: Evidence set $\mathbb{E} = \{D_1, \ldots, D_n\}(n \geq 2)$; 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 **then**
13: **Return** True
14: **else**
15: **Return** False
16: **end if**

A. Divergence-Based Collective Anomaly Detection

From section III we know that statistical divergence only provides a distance between two or more distributions. In a set of data collections, we can only draw a complete graph where nodes denote data collections and edges refer to the divergence between two connected nodes. From the graph we can find some points that have apparently larger distances with most of other points and return them as anomalies. This may work if anomalous nodes do not compose a large proportion. However the procedure will be too complicated to work out with large amounts of data collections.

Otherwise we need a frame of reference that provides absolute coordinates rather than the relative ones. Therefore, we should keep an evidence set $\mathbb{E}$ which consists of several correct distributions that have already been given. Then the real distribution can be estimated by averaging all distributions in $\mathbb{E}$. This process is similar to the parameter estimation within a certain sample set. Let M denote the estimated population distribution, then we can measure the statistical distance with respect to M for every distribution D inside and outside $\mathbb{E}$, yielding absolute distances.

Fig. 9 in section V-B demonstrates the result of the above process. Red dots refer to the distances calculated from normal data collections, blue and green ones are from click farmed data collections. Clearly, distances of normal data collections assembles together around a small value while anomalous ones lay around a larger distance value. Intuitively, a simple gaussian classifier can be suitable here to discover anomalous data points. The basic framework of the algorithm is shown in Algorithm 1.

Let $\mathbb{S}$ be set of the daily transaction collections, if $\mathbb{E}$ is already given, we can run the algorithm with $\mathbb{E}$ and $\forall D \in \mathbb{S}$ and discover the anomalies. JSD here cannot be replaced by KLD since entries in P' are not necessarily defined in M. 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 viable 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 very large. For example, suppose $P(1) = P(2) = P(3) = \frac{1}{3}$ and $P'(1) = \frac{1}{6}, P'(2) = \frac{1}{3}, P'(3) = \frac{1}{2}$, then $JSD(P||P') \approx 0.023$ and $JSD(P, P, P, P') \approx 0.017$.

According to the property of statistical divergence, we can infer that the true distribution of JSDs calculated form normal data collections are close to but not exactly a gaussian distribution $N(\mu, \sigma)$ since for each point, there are both definite upper and lower bounds instead of infinities. Therefore, μ should be slightly larger than zero. The reason why μ cannot be exactly zero is hidden in line 4. For $\mu = 0$ if and only if $J_i = 0$ for $\forall i = 1, \ldots, n$, which is equivalent to $P_i = P_j$ for $\forall i, j = 1, \ldots, n$. And, $\lim_{n \to +\infty} P(P_i = P_j, \forall i, j = 1, \ldots, n) = 0$. Thus it is highly unlikely for real world data sets to have $u = 0$.

B. Distribution Histogram

Not all data collections can be assumed subject to known distributions. Parameter approximation will probably lead to errors far beyond our expectation. For example, the distribution of vehicle volume in the highway may be of the shape of two connected gaussian curves and does not correspond to any of the prevalent distribution models. 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 on any sampled data. But 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 an 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

$$l = c\sigma k^{-0.2} \tag{6}$$

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

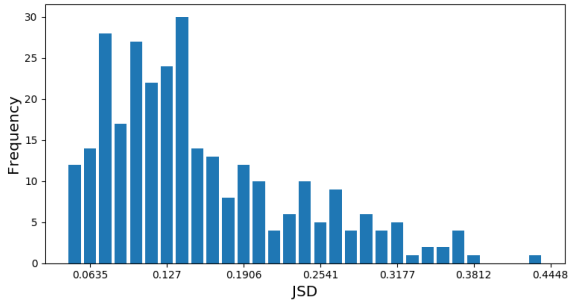

Fig. 2. Distribution of JSD values for everyday data collection. It can be approximately regarded as a gaussian distribution.

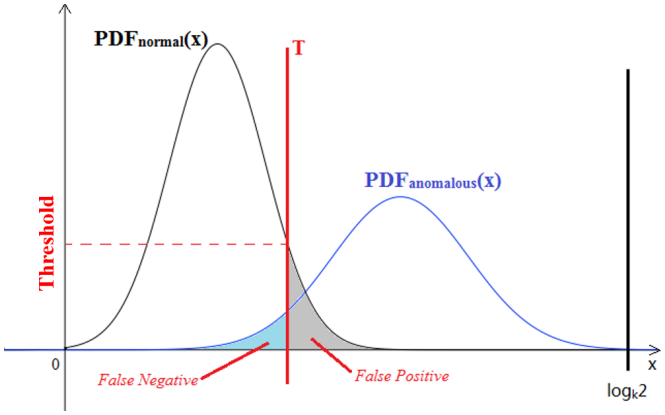

Fig. 3. Threshold can be determined either by a probability density value, or a radius from the centre. In the scenario shown in the figure, the threshold can be determined by the position between two centres of the distribution, denoted as "T" here. This form of threshold can also be applied to other types of distributions even there is no intersection between these two.

1.05). For data sets with a large number of elements, a random sampling method, such as Monte-Carlo method, can be applied to speed up the estimation procedure.

C. 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 specific analysis in the certain scenario, manual observation and tuning of parameters, which involves lots of human labour. The rule of "3σ" can be used to automatically determine a threshold. But as a rigid metric, it is merely an estimation of a suitable boundary considering average situations, which is far from optimal when concrete data is provided. It would be either higher than the optimum if anomalous data lies far away from the normal cluster, or lower than the optimum if the anomalies sit close to the cluster centre.

However, applying divergence as the distance measurement among data collections provides a fine property. That is, divergences of normal data collections assemble together, forming a quasi-gaussian distribution. And those of anomalies also form a quasi-guassian distribution which lies in the right-hand-side to the other one on the real number axis. As a result of further observation, however, true distribution of both JSDs derived from normal and anomalous data collections may differ a little more from the standard gaussian than the expected estimation error(Fig. 2). That is due to the unknown randomness within real world data. Few assumptions can be applied in real world data sets, let alone data volume is sometimes relatively low. This topic is out of the domain discussed in this paper and we here only introduce the technique instead of the specific distribution model. However, in the scenarios that better models can be calculated, distributions here can be replaced to produce better results. For the simplicity of our proposal, we assume the distributions of JSDs to be gaussian.

As shown in Fig 3, the black curve($PDF_{normal}(x)$ or in short $PDF_n(x)$) displays the probability density function(PDF) fitting those JSDs calculated from normal data collections; the blue curve($PDF_{anomalous}(x)$ or in short $PDF_a(x)$) displays the PDF derived from anomalous data collections. Threshold is chosen to minimize total errors(both false negative and false positive).

Suppose:

$$PDF_n(x) \approx N(\mu_n, \sigma_n) \tag{7}$$
$$PDF_a(x) \approx N(\mu_a, \sigma_a) \tag{8}$$

Then the optimal threshold T is:

$$\begin{aligned} T &= \arg\min_T \int_0^T PDF_a(x)dx + \int_T^{log_k 2} PDF_n(x)dx \\ &\approx \arg\min_T \int_0^T \frac{e^{-\frac{(x-\mu_a)^2}{2\sigma_a^2}}}{\sqrt{2\pi}\sigma_a}dx + \int_T^{log_k 2} \frac{e^{-\frac{(x-\mu_n)^2}{2\sigma_n^2}}}{\sqrt{2\pi}\sigma_n}dx \\ &= \frac{\mu_n\sigma_a + \mu_a\sigma_n}{\sigma_a + \sigma_n} \end{aligned} \tag{9}$$

According to equation(9), 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(9) implicates 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:

$$T = \arg\min_T \alpha \int_0^T PDF_a(x)dx + (1-\alpha)\int_T^{log_k 2} PDF_n(x)dx$$
$$\approx \arg\min_T \alpha \int_0^T \frac{e^{-\frac{(x-\mu_a)^2}{2\sigma_a^2}}}{\sqrt{2\pi}\sigma_a}dx + (1-\alpha)\int_T^{log_k 2} \frac{e^{-\frac{(x-\mu_n)^2}{2\sigma_n^2}}}{\sqrt{2\pi}\sigma_n}dx \quad (10)$$

Where α is the anomaly probability. Equation(10) is much more complicated. But it can be reduced to a quadratic equation. Section V-D compares some estimations for optimal threshold under a real world circumstance. It shows the difference among calculators and that equation(10) outperforms others when $PDF_n(x)$ and $PDF_a(x)$ are actually gaussian.

D. Dynamic DCAD

In Algorithm 1, there is a hidden assumption that it regards every data collection to be sampled from a static environment. That is to say the seller will never change his or her products and customer volumes always remain the same. In most cases, real world data is always in the process of evolution and fluctuation. Online shops are often in the process of expanding or dwindling. Thus the sales distribution will slightly change as time goes on. 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. And considering discussion in section IV-B and IV-C, a dynamic version is proposed in Algorithm 2.

Algorithm 2 uses $\mathbb{E}_N$ and $\mathbb{E}_A$ 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 version. 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.

Note that line 2 and 5 can be replaced by other estimation methods if better assumptions can be made. Line 16 and 17 can be replaced by other models for more accuracy. 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.

V. Evaluation

The algorithm was implemented and interpreted in Python 3.6. All experiments was tested on Ubuntu 17.04. In the following experiments, we would like to figure out:

1) How to emulate different types of click farming?
2) Is the analysis in section II-B and IV-A applicable and accurate in real world data sets.
3) Is the technique efficient against click farming? And How sensitive will the classifier be confronting different magnitude of click farming?
4) What is the performance of different adaptive thresholds?

These four questions are answered in the following subsections respectively.

Algorithm 2 Dynamic Classification

Input: Normal evidence set $\mathbb{E}_N = \{D_{N_1}, \ldots, D_{N_n}\}$ by given order; Anomalous evidence set $\mathbb{E}_A = \{D_{A_1}, \ldots, D_{A_m}\}$ 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: $\mathbb{E}_N \leftarrow \mathbb{E}_N \backslash \{D_{N_0}\} \cup \{D'\}$
21: **Return** False
22: **else**
23: $\mathbb{E}_A \leftarrow \mathbb{E}_A \backslash \{D_{A_0}\} \cup \{D'\}$
24: **Return** True
25: **end if**

A. Methodology

We adopted a data set containing Taobao online sellers' transaction records[1] provided by Alibaba Tian Chi big data competition where all records are collected from the real world business scenarios and are desensitized. 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 Examples

User ID	Seller ID	Payment Time
10523185	1629	2015-11-11 17:00:00
13799128	1862	2016-07-05 12:00:00
22127870	1862	2015-12-25 17:00:00
⋮	⋮	⋮

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

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

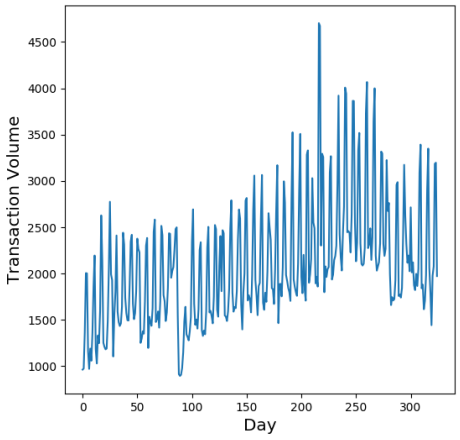

Fig. 4. Changing of the daily sales volume shows that environment of online sales has been changing all the time. Business transactions shall be checked with the dynamic algorithm.

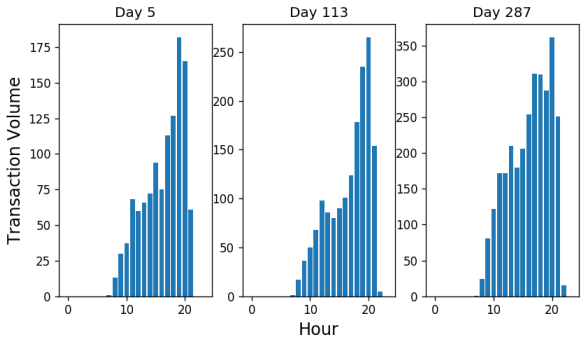

Fig. 5. We selected 3 days randomly and drew sales distribution by counting hourly volume. Although sales volume has changed from day to day, the shape of the distribution remain almost alike.

divided into 325 collections, each containing records in one day. There are some days with no records because the online shop temporarily closed in those days.

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 section II-B. We adopted two prevalent types of click farming: *centralized click farming* and *equalized click farming*. Centralized click farming emulates the scenarios that a group of people are called out and assigned tasks by a casual leader that does not care much about the click farm timing. A significant feature of this approach is that the cheating transactions usually assemble together in a short period of time. Equalized click farming emulates the circumstances that click farms are arranged by some well programmed applications or teams carefully managed and strictly commit transactions according to a timetable. Thus the transaction distribution may not vary too much with and without click

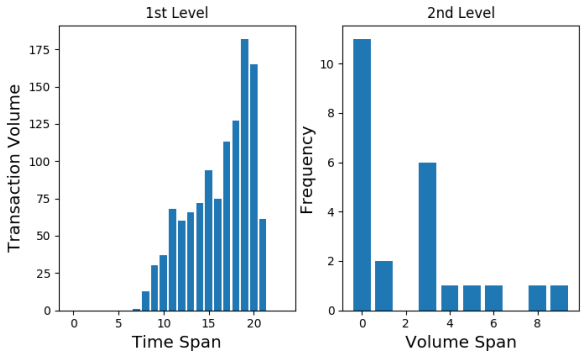

Fig. 6. Example of 1st and 2nd Level Histogram

farming. Usually, the click farmed transactions are several times more than the volume it original has, if the seller hires a group of organized workers. In our experiments, we use ν to denote the magnitude coefficient of click farming. Hence $|D_{anomalous}| = (1+\nu)|D_{normal}|$. In the following experiments without extra illustration, we adopt $\nu = 1$.

To emulate centralized click farming, we randomly inserted some gaussian-distributed transactions in the chosen collection. As for emulating the equalized click farmers, 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 aligned to time spans. The second level is to draw a histogram on the sub-volumes in each time span. For example, as 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: $\mathbb{F} = \{0, 0, 0, 0, 0, 0, 0, 1, 13, 30, 37, 68, 60, 66, 72, 94, 75, 113, 127, 182, 165, 61, 0, 0\}$. Then, counting $\mathbb{F}$ with a step size 20 gives the 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 input collection 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 or false. Several metrics were explored in the following experiments.

B. Experiments 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 each transaction was committed. Thus, the histogram was drawn by counting hourly volume of each collection. For $\alpha = 0.2$, we selected 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(5) in Table V was employed as the threshold calculator. The

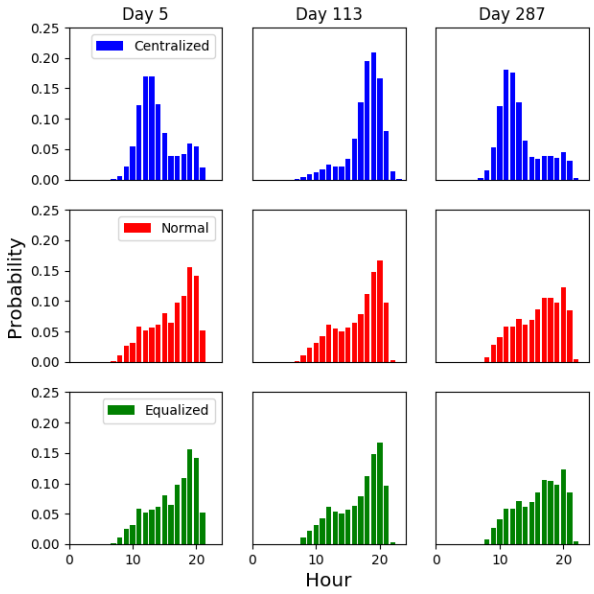

Fig. 7. 1st level histogram of day 5, 113 and 287, each day in a column. Distributions after centralized and equalized click farming are in 1st and 3rd row correspondingly. And the original distributions are shown in 2nd row.

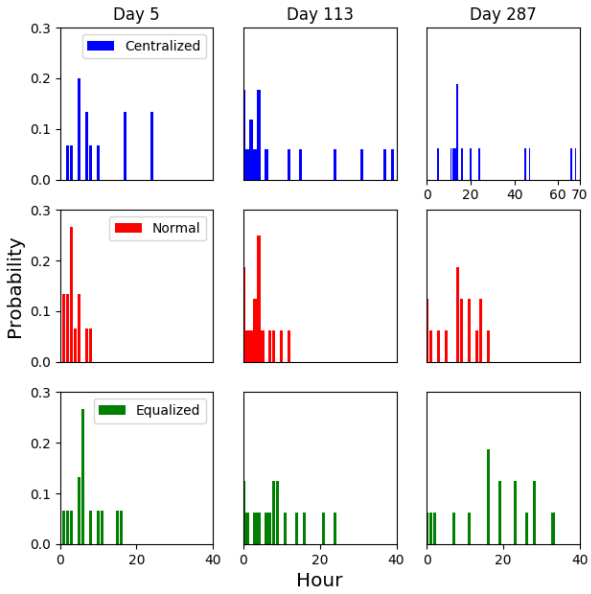

Fig. 8. 2nd level histogram of day 5, 113 and 287, each day in a column. Distribution after centralized click farming in day 287 uses another time scale since it deviates a lot from others.

results are shown in Table II, where *true positive rate*(TPR), *false positive rate*(FPR) and *accuracy*(ACC) were recorded.

TABLE II
CLASSIFICATION RESULTS ON RAW DATA

	Centralized			Equalized		
Level	TPR	FPR	ACC	TPR	FPR	ACC
1st	100.00	4.05	96.72	26.64	28.03	62.49
2nd	88.01	7.68	91.41	83.89	15.78	84.07

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 histograms, 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, the difference between normal and anomalous collections can be observed more intuitively.

C. Experiment on Synthetic Data

As is put in the former section, every single record in transaction collections gives only 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.

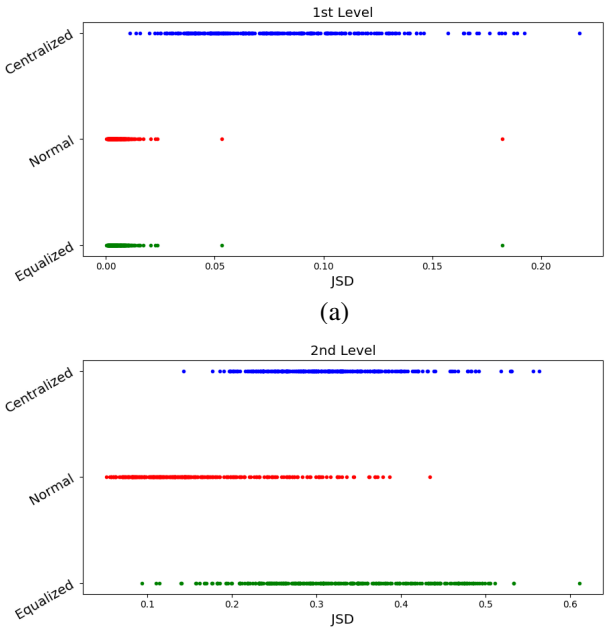

Fig. 9. This figure shows the JSD values of 1st and 2nd level histogram in each day in three different version: normal, centralized click farmed and equalized click farmed. Both (a) and (b) shows that centralized ones are usually larger than normal ones and most of them clearly distinguishable. In (a), 1st level histogram cannot tell anything different between normal ones and equalized ones. But in (b), 2nd level histograms map the equalized click farmers to the right side of normal ones.

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(6) 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(6) 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 $\mathbb{E}_N$ and $\mathbb{E}_A$ the same configuration as before. Results are shown in Table III and IV.

TABLE III
CLASSIFICATION ON 1ST LEVEL HISTOGRAM

	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
CLASSIFICATION ON 2ND LEVEL HISTOGRAM

	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 in them 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 not relevant directly to TPR or FPR. When α was increasing, there were less data collections added into $\mathbb{E}_N$. Thus the update of $\mathbb{E}_N$ became slower and FPR went higher. On the contrary, TPR became higher as α increased. Moreover, since $|\mathbb{E}_N| > |\mathbb{E}_A|$, it needs more collection instances for $|\mathbb{E}_N|$ then that for $|\mathbb{E}_A|$ to keep up to the trend. Therefore, it was faster of FPR to increase than that of TPR. 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. 10 shows the ROC curve of the gaussian classifier dealing with centralized click farming with 1st level histograms

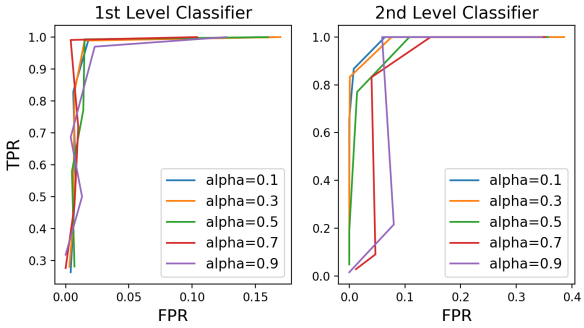

Fig. 10. ROC curve of 1st and 2nd level classifier given different values of α. The performance of the classifier does not change much for different error rates.

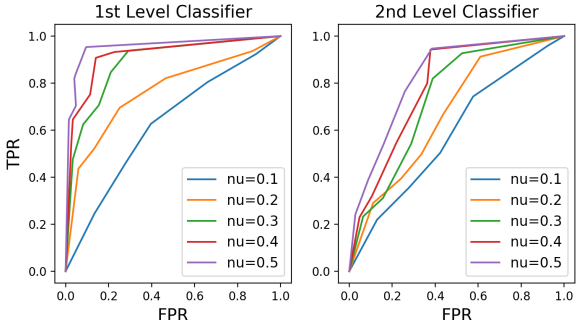

Fig. 11. ROC curve of different magnitude of click farming. In real world scenarios, ν is normally larger than 1.

and equalized click farming with 2nd level histograms. The curves are pretty close to the upper left corner of the diagram and size under the curves are very large. For different α's, the classifier can efficiently distinguish normal and anomalous data collections.

In Fig. 11, we tested the classifier with different magnitude of centralised click farming. In former experiments, $\nu = 1$. If ν is smaller, then it will be harder for the classifier to identify anomalous data collections since the anomalous distribution will be closer to normal ones. It can be concluded from the figure that the 1st level classifier can be efficient even when $\nu = 0.3$. However the 2nd level classifier is less sensitive. That is because the 1st level histogram provides more probability entries than 2nd level histograms do. Data collections are compressed from 1st to 2nd level. But for real world click farming behaviours(usually, $\nu > 1$) both classifiers can be efficient enough.

D. Threshold Optimization

Section IV-C mentions only two equations for calculating adaptive thresholds. Table V gives the comparison of more equations out of similar considerations. From the aspect of weighted average(i.e. equation(1)) and taken α into consideration, we composes several other formulas with different operation on α and σ, yielding other possible balanced points within the interval $[\mu_N, \mu_A]$.

TABLE V
F1 SCORES OF DIFFERENT THRESHOLD CALCULATORS

	Equation	Synthetic	Gaussian
1	$T = \frac{\mu_n\sigma_a + \mu_a\sigma_n}{\sigma_a + \sigma_n}$	1.0588	0.5810
2	$T = \frac{\mu_n\sigma_a(1-\alpha) + \mu_a\sigma_n\alpha}{\sigma_a(1-\alpha) + \sigma_n\alpha}$	0.7688	0.3515
3	$T = \frac{\mu_n\sigma_a\sqrt{1-\alpha} + \mu_a\sigma_n\sqrt{\alpha}}{\sigma_a\sqrt{1-\alpha} + \sigma_n\sqrt{\alpha}}$	0.8158	0.4510
4	$T = \frac{\mu_n\sigma_a ln(1-\alpha) + \mu_a\sigma_n ln\alpha}{\sigma_a ln(1-\alpha) + \sigma_n ln\alpha}$	1.5227	0.5523
5	$T = \frac{\mu_n\sigma_a\sqrt{-ln(1-\alpha)} + \mu_a\sigma_n\sqrt{-ln\alpha}}{\sigma_a\sqrt{-ln(1-\alpha)} + \sigma_n\sqrt{-ln\alpha}}$	**2.6013**	0.5791
6	$T = \frac{\mu_n\sigma_a ln\sqrt{1-\alpha} + \mu_a\sigma_n ln\sqrt{\alpha}}{\sigma_a ln\sqrt{1-\alpha} + \sigma_n ln\sqrt{\alpha}}$	1.4155	0.5578
7	$T = \frac{\mu_n\sqrt{\lvert ln[\sigma_a(1-\alpha)]\rvert} + \mu_a\sqrt{\lvert ln(\sigma_n\alpha)\rvert}}{\sqrt{\lvert ln[\sigma_a(1-\alpha)]\rvert} + \sqrt{\lvert ln(\sigma_n\alpha)\rvert}}$	2.2730	0.5600
8	$T = \frac{\mu_n ln\sqrt{\sigma_a(1-\alpha)} + \mu_a ln\sqrt{\sigma_n\alpha}}{ln\sqrt{\sigma_a(1-\alpha)} + ln\sqrt{\sigma_n\alpha}}$	2.0753	0.2808
9	$T = \underset{\mu_n \le T \le \mu_a}{\arg\min}\ \alpha \cdot \int_0^T PDF_a(x)dx + (1-\alpha) \cdot \int_T^{log_k 2} PDF_n(x)dx$	1.9300	**0.6072**

We tested those calculators on synthetic data set with $\alpha = 0.1, \nu = 1$. Testing each level of classifier and each type of anomalies, we got 4 F1 scores and added them up. Equation 5 gives best performance. Then those equations were tested on large amount of gaussian distributed data instances. 90K and 10K data points were sampled from 500 pairs of random gaussian distributions which represent normal and anomalous distributions respectively, yielding 500 F1 scores for each equation. We compared the average score and discovered that equation 9 gives the best result.

Equation 9 gives best classification result for two gaussian distributions but does not outperform others in real world data. Part of the reasons are mentioned in section IV-C. Another reason is that the size of evidence sets are too small, that the gaussian distribution may not be well approximated.

VI. CONCLUSION & FUTURE WORK

This paper proposed an dynamic collective anomaly detection technique with adaptive threshold, which helps detect data manipulations in modern data pipelines and data centres. Different from existing algorithms designed for collective anomalies, our approach employs statistical distance as the similarity measurement, mapping data collections to a restricted one dimensional space. We explored several technical points involved in the design of the algorithm and performed a thorough experiment to test its efficiency. It showed that the algorithm can efficiently discover anomalies within the data collections and the classifier is sensitive enough toward real world data manipulations.

However, the modelling of the JSD distributions is not precise enough. The efficiency of different adaptive thresholds should be tested on different real world circumstances to get a more complete view of their performances. Moreover, a comparison with other collective detection algorithm should be adopted in the future to get an performance overview of similar algorithms. Since the algorithm adapts to the slowly evolving feature, when attacks disguise itself in the long run as if it was one of the changing features, the technique may fail to detect. In this case, other techniques may be applied as assistances.

ACKNOWLEDGMENT

REFERENCES

[1] (2017) Big data statistics and facts for 2017. [Online]. Available: https://www.waterfordtechnologies.com/big-data-interesting-facts/

[2] (2017). [Online]. Available: https://raygun.com/blog/cost-of-software-errors/

[3] K. Pawar, A. Jadhav, P. Hande, B. Ambadkar, and R. Warade, "Software data reduction technique for bug reduction problem," *International Journal*, vol. 4, no. 4, 2016.

[4] T. Xu and Y. Zhou, "Systems approaches to tackling configuration errors: A survey," *ACM Computing Surveys (CSUR)*, vol. 47, no. 4, p. 70, 2015.

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

[6] M. A. Rassam, M. A. Maarof, and A. Zainal, "Adaptive and online data anomaly detection for wireless sensor systems," *Knowledge-Based Systems*, vol. 60, pp. 44–57, 2014.

[7] H. Herodotou, B. Ding, S. Balakrishnan, G. Outhred, and P. Fitter, "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.

[8] F. Schuster, M. Costa, C. Fournet, C. Gkantsidis, M. Peinado, G. Mainar-Ruiz, and M. Russinovich, "Vc3: Trustworthy data analytics in the cloud using sgx," in *Security and Privacy (SP), 2015 IEEE Symposium on*. IEEE, 2015, pp. 38–54.

[9] TowerData. (2013) 4 steps to eliminating human error in big data. [Online]. Available: http://www.towerdata.com/blog/bid/113787/4-Steps-to-Eliminating-Human-Error-in-Big-Data

[10] Is data manipulation the next step in cyber crime. [Online]. Available: https://www.cloudmask.com/blog/is-data-manipulation-the-next-step-in-cybercrime

[11] (2016). [Online]. Available: https://www.cnbc.com/2016/03/09/the-next-big-threat-in-hacking–data-sabotage.html

[12] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM computing surveys (CSUR)*, vol. 41, no. 3, p. 15, 2009.

[13] A. L. Goldberger *et al.*, "Components of a new research resource for complex physiologic signals, physiobank, physiotoolkit, and physionet, american heart association journals," *Circulation*, vol. 101, no. 23, pp. 1–9, 2000.

[14] R. Fujimaki, T. Yairi, and K. Machida, "An approach to spacecraft anomaly detection problem using kernel feature space," in *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*. ACM, 2005, pp. 401–410.

[15] L. Cao, D. Yang, Q. Wang, Y. Yu, J. Wang, and E. A. Rundensteiner, "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.

[16] L. Cao, Y. Yan, C. Kuhlman, Q. Wang, E. A. Rundensteiner, and M. Eltabakh, "Multi-tactic distance-based outlier detection," in *Data Engineering (ICDE), 2017 IEEE 33rd International Conference on*. IEEE, 2017, pp. 959–970.

[17] M. M. Masud, Q. Chen, L. Khan, C. C. Aggarwal, J. Gao, J. Han, A. Srivastava, and N. C. Oza, "Classification and adaptive novel class detection of feature-evolving data streams," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 7, pp. 1484–1497, 2013.

[18] Y. Li, Q. Li, J. Gao, L. Su, B. Zhao, W. Fan, and J. Han, "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] J. Shao, Z. Ahmadi, and S. 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.

[20] X. Zhang, W. Dou, Q. He, R. Zhou, C. Leckie, R. Kotagiri, and Z. Salcic, "Lshiforest: A generic framework for fast tree isolation based ensemble anomaly analysis," in *Data Engineering (ICDE), 2017 IEEE 33rd International Conference on*. IEEE, 2017, pp. 983–994.

[21] J. Yin and J. 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.

[22] Y. Zhu and D. 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.

[23] 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.

[24] P. K. Chan and M. V. Mahoney, "Modeling multiple time series for anomaly detection," in *Data Mining, Fifth IEEE International Conference on*. IEEE, 2005, pp. 8–pp.

[25] S. Budalakoti, A. N. Srivastava, R. Akella, and E. Turkov, "Anomaly detection in large sets of high-dimensional symbol sequences," 2006.

[26] N. 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.

[27] C. Warrender, S. Forrest, and B. 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.

[28] D. 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.

[29] R. Yu, X. He, and Y. Liu, "Glad: group anomaly detection in social media analysis," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 10, no. 2, p. 18, 2015.

[30] Iresearch. (2016). [Online]. Available: http://report.iresearch.cn/content/2016/11/265551.shtml

[31] M. Zhao, "On the phenomenon of click farming in taobao," *CHINA BUSINESS*, no. 12, pp. 31–33, 2016.

[32] Z. Yan, "Report of click farming phenomenon in taobao–protection for consumers' right to know in online shopping," *FaZhi Yu JingJi*, vol. 20, pp. 195–196, 2015.

[33] T. L. Mirror. (2016). [Online]. Available: http://tech.163.com/15/0405/14/AMENOJ7D00094ODV.html

[34] (2015). [Online]. Available: http://www.ebrun.com/20150906/147724.shtml