Online E-Commerce Fraud: A Large-scale Detection and Analysis

Haiqin Weng [1], Zhao Li [2], Shouling Ji [1], Chen Chu [2], Haifeng Lu [2], Tianyu Du [1], Qinming He [1],

[1] *College of Computer Science, Zhejiang University, Hangzhou, China*
{hq_weng, sji, zjradty, hqm}@zju.edu.cn

[2] *Alibaba Group, Hangzhou, China*
{lizhao.lz, chuchen.cc, haifeng.lhf}@alibaba-inc.com

***Abstract*—Nowadays, e-commerce has become prevalent worldwide. With the big success of e-commerce, many malicious promotion services also rise: with the goal of increasing sales, malicious merchants attempt to promote their target items by illegally optimizing the search results using fake visits, purchases, etc. In this paper, we study the fraud detection problem on large-scale e-commerce platforms. First, we develop an efficient and scalable *AnTi-Fraud system* (ATF) to detect e-commerce frauds for large-scale e-commerce platforms, and implement it in parallel on a large-scale computing platform, called Open Data Processing Service (ODPS). Then, we evaluate ATF using two real large-scale e-commerce datasets (with tens of millions users and items). The results demonstrate that both the precision and the recall of ATF can achieve $0.97+$, which suggests that ATF is very effective. More importantly, we deploy ATF on the `Taobao` platform of `Alibaba`, which is one of the world's largest e-commerce platforms. The evaluation results show that ATF can also achieve an accuracy of 98.16% on `Taobao`, which again suggests that ATF is very effective and deployable in practice. Our study in this paper is expected to shed light on defending against online frauds for practical e-commerce platforms.**

I. Introduction

Background. With the rapid development of information technologies, *e-commerce* now becomes prevalent worldwide. E-commerce platforms efficiently connect customers with factories, stores, and third-party merchants, and serve billions of users with numerous products and services (denoted by *items* in this paper) everyday, providing them a convenient, fast, and reliable manner of shopping, service acquisition, reviewing, comment feedback, etc. For instance, `Ebay` is reported to have more than 164 million annual active customers in 2016[1], `Amazon` is reported to have 310 million annual active customers in 2016[2], and `Taobao` (belongs to the `Alibaba Group`) is reported to have 443 million annual active customers in 2016[3]. These e-commerce platforms can serve the customers billions of items, which make people's life significantly convenient.

Due to the incredibly huge amount of online items, the search engine of e-commerce platforms is the major entrance to access the items for users. To find items of interest, people usually query the search engine using the keywords of those items, and then view or further purchase them. Such subsequent browsing and purchase referred by the search results account for the main incoming traffic volume of an item.

In reality, people are inclined to view or buy the items that are listed in the front of the search results, have been purchased by many other people, and/or have high review scores [1]. Therefore, with the goal of increasing sales, malicious merchants (adversaries) attempt to promote their target items by illegally optimizing the search results using fake visits, purchases, and/or feedback. For instance, adversaries may hire a group of human labors to visit their target items frequently, aiming to create a false impression that those items are popular among customers. In this paper, for convenience, we name the malicious e-commerce promotions as *e-commerce frauds*, the maliciously promoted items as *fraud items* and the users who conduct malicious promotions as *dishonest users* (e.g., the human labors being hired to visit a fraud item).

E-Commerce Fraud Detection. Malicious promotion is very harmful to the e-commerce ecosystem in many perspectives, e.g., it harms the online advertising system and causes unfair competition; it provides fake information to customers and can further mislead them to make improper decisions; and so on [2]. As reported in [2][3], malicious promotions have caused hundreds of millions dollars of loss worldwide. To make things worse, malicious promotion shows an increasing trend recently [2][3].

However, to our knowledge, it is seldom to see a dedicated and efficient fraud detection system for large-scale online e-commerce platforms. Existing fraud detection techniques are either designed for other application domains, e.g., search engine click fraud [4], tax fraud [5], and phone fraud [6], which are improper for e-commerce fraud detection, or not sufficiently scalable or robust to be applied on large-scale online e-commerce platforms of billions of users and items. On the other hand, e-commerce fraud detection is a challenge problem since (1) *the scalability issue:* fraud items often hide in billions of benign items, which makes it difficult, if not impossible, to identify them using algorithms with high computational complexity; (2) *the efficiency and robustness issue:* fraud items are usually promoted using various and dynamically updated strategies, and the promotions are intensionally performed in a manner of mimicking benign users'

*Haiqin Weng and Zhao Li are co-first authors.
*Shouling Ji is the corresponding author.

[1]https://static.ebayinc.com
[2]https://www.statista.com/topics/846/amazon/
[3]https://www.alibabagroup.com

behaviors. Thus, effectively characterizing and modeling e-commerce frauds is a nontrivial problem. In summary, to facilitate the healthy development of e-commerce, it is important to understand those malicious e-commerce promotions, i.e., *online e-commerce frauds*, and further defend against them by developing an efficient, robust, and scalable fraud detection system.

Our Methodology. Aiming at developing an efficient, scalable, and robust fraud detection system for large-scale online e-commerce platforms, we present an *AnTi-Fraud* system (ATF) in this paper. ATF mainly consists of three components: *preprocessor*, *Graph-Based Detection module* (GBD), and *Time Series based Detection module* (TSD). The preprocessor is used for raw data processing and preparing necessary data for GBD and TSD. Specifically, it will construct a *user-item bipartite graph* for GBD and each item's *traffic time series* for TSD. Based on the user-item bipartite graph and a small set of confirmed dishonest users, GBD assigns each item a fraud score via a propagation algorithm. Then, it determines the fraud items based on their fraud scores. TSD is designed based on the observation that when a new fraud pattern appears or a new item becomes a fraud item, the traffic time series of the new fraud item is likely to exhibit differently from the benign items. Therefore, it first makes a hypothesis that the traffic time series of each item follows a *mixture Poisson distribution*. Then, it derives an *abnormal score* for the traffic time series of each item, and determines the fraud items based on the abnormal scores. In our design, GBD and TSD have different focuses: GBD emphasizes detecting fraud items that are promoted using similar strategies with known patterns, while TSD focuses on detecting new fraud items and those are generated following new promotion strategies.

Implementation and Evaluation. We implement ATF on the Open Data Processing Service (ODPS) platform provided by `Alibaba`. Leveraging two large-scale e-commerce datasets D_1 (47,013,511 users and 2,499,125 items) and D_2 (13,676,596 users and 503,293 items), we validate the performance of ATF. On D_1, ATF achieves a precision of 0.9764 and a recall of 0.9785, and on D_2, ATF achieves a precision of 0.9749 and a recall of 0.9872. This suggests that ATF is very effective.

Deployment. To evaluate the performance of ATF on real e-commerce platforms, we deploy ATF on the `Taobao` platform of `Alibaba`, which is considered as one of the world's largest online e-commerce platforms. According to the report from `Alibaba`, it has 443 million annual active customers in 2016 and serves customers billions of items. After running ATF, we report the detection results of ATF to `Alibaba`. Through the analysis of domain experts, `Alibaba` confirms that 98.16% of the results are fraud items, which indicates that ATF is also very effective on real e-commerce platforms. We further examine the scalability of ATF on `Taobao`, which demonstrates that ATF is scalable when dealing with the data over $O(100$ million$)$-scale of active users and $O($billion$)$-scale of items. This implies that ATF can be applied on real large-scale e-commerce platforms.

Contributions. In summary, we make the following contributions in this paper. (1) *AnTi-Fraud (ATF) System.* We develop an efficient and scalable large-scale e-commerce fraud detection system, named ATF. ATF can effectively detect the e-commerce frauds generated by both existing and new malicious promotion strategies. (2) *Implementation and Evaluation of ATF.* We implement ATF on the ODPS platform provided by `Alibaba`. Leveraging two real large-scale e-commerce datasets, we validate the performance of ATF, which achieves both high precision and high recall. (3) *Deployment.* We deploy ATF on the `Taobao` platform of `Alibaba`, which is one of the world's largest online e-commerce platforms. Through evaluation, we demonstrate that ATF is also very effective and scalable in practical scenarios.

II. Anti-Fraud System

In this section, we first give an overview of ATF, followed by its detailed design and implementation.

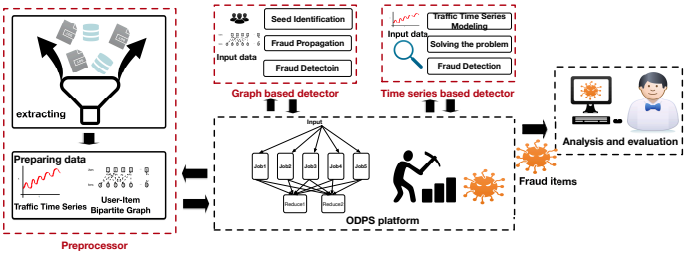


Fig. 1. Architecture of ATF

A. Design of ATF

Fig. 1 illustrates the architecture of ATF, which mainly consists of three components: *preprocessor*, *Graph-Based Detection module* (GBD), and *Time Series based Detection module* (TSD). We present the main function and design of each component below.

1) Preprocessor: The main function of the preprocessor is for data preprocessing. When the raw log files of the online e-commerce activities come to ATF, the preprocessor first filters the noise in the data, e.g., the click records generated by unregistered users. Then, it extracts *user click records*, which are considered as the most fundamental behavior for online e-commerce activities (e.g., browsing, shopping, and reviewing) from those log files. Based on user click records, the preprocessor prepares necessary data for the following fraud detection. Specifically, it will construct a *user-item bipartite graph* for the GBD module and each item's *time series of incoming traffic* for the TSD module.

User-Item Bipartite Graph. Let $U = \{U_i | i = 1, 2, \cdots\}$ be the e-commerce user set and $I = \{I_i | i = 1, 2, \cdots\}$ be the e-commerce item set. Then, based on the user click records, we can construct a user-item bipartite graph $G = (U \cup I, E)$, where $E = \{E_{i,j} | \exists U_i \in U, I_j \in I$, and a click record such that U_i clicked $I_j\}$. Without loss of generality, we assume that

G is connected in this paper[4]. We take G as the input for the GBD module.

Traffic Time Series. Given a time slot t (e.g. one day) and an item $I_i \in I$, we define the total number of clicks made on I_i by users in U within t as the *traffic volume* of I_i in that time slot, denoted by $V_i(t)$. Then, given a time window T that consists of n time slots, we define the *traffic time series* of I_i in T as $T_i = \{V_i^1(t), V_i^2(t), \cdots, V_i^n(t)\}$, where $V_i^j(t)$ is the traffic volume of I_i in the j-th time slot. Based on the extracted user click records and a predefined time slot t, the preprocessor can construct the traffic time series for each item. We take the traffic time series of all the items as the input for the TSD module.

2) Graph-based Detection (GBD): In this subsection, we discuss the GBD module, which is designed for performing fraud detection leveraging the structural and behavioral characteristics of e-commerce frauds. For each malicious promotion task, it is usually accepted and finished by a group of dishonest users. Therefore, those dishonest users and fraud items tend to have strong connections with respect to the user-item click relationship, as observed in existing empirical analysis [3]. Motivated by this intuition, we design GBD as follows: leveraging the user-item bipartite graph G constructed by the preprocessor and a small set of identified dishonest users, namely *seeds*, it assigns each item a *fraud score* via a propagation algorithm. Then, it determines the fraud items in terms of its associated fraud score.

Seed Identification. The first step of GBD is to identify a small set of dishonest users, denoted by $S = \{U_i | U_i$ is a dishonest user$\}$, to bootstrap the fraud detection. In practice, this step can be finished through multiple methods: (1) *Identifying dishonest users with the help of domain experts.* For instance, in some fraud scenarios, "professional" dishonest users may regularly visit/click several thousands of items in a short time. Such kind of dishonest users can be easily identified by domain experts. In some other fraud scenarios, the items of an online store are abnormally visited by a small group of suspicious users in a short time. Those suspicious users can be further analyzed by domain experts to see if they are dishonest users; (2) *Identifying dishonest users leveraging cross-platform analysis and user alignment techniques* [7][3]. As we explained in Section I, many dishonest users accept promotion tasks on the malicious service platforms (e.g., `Diyishuadan` [5]). Therefore, we can also collect the published malicious tasks, and further leverage cross-platform analysis techniques and user alignment techniques to identify dishonest users (usually, we can also identify some fraud items in this way) [7][3]; (3) *Identifying dishonest users using Sybil detection techniques* [8]. Since many e-commerce Sybil users themselves are dishonest users [8], we can identify them using Sybil detection techniques along with the knowledge from domain experts.

[4]In the case that G is disconnected, we can apply our fraud detection algorithm to each connected component of G directly.

[5]http://diyishuadan.tm12.cn/jd/rw_lb.php

Fraud Propagation. After obtaining the seed set S, our next step is to assign each item $I_i \in I$ a *fraud score*, denoted by $p(I_i)$, which indicates the probability that I_i is a fraud item. We also assign each user $U_i \in U$ a *dishonest score*, denoted by $p(U_i)$, which indicates the probability that U_i is a dishonest user. Initially, we set (1) $\forall U_i \in S, p(U_i) = 1$; (2) $\forall U_i \in U \setminus S, p(U_i) = 0$; and (3) $\forall I_i \in I, p(I_i) = 0$.

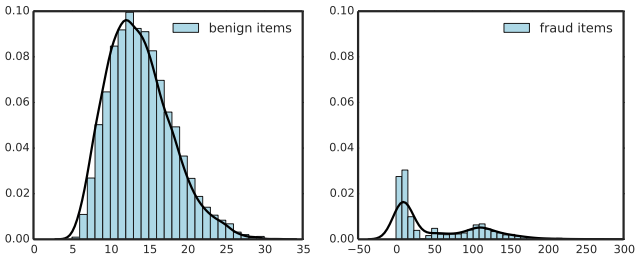
Then, starting from the seed set S and leveraging the user-item bipartite graph G, we iteratively compute the fraud score (resp., dishonest score) of each item in I (resp. each user in U) using the information propagation algorithm HITS [9]. Specifically, for $p(I_j), I_j \in I$ and $p(U_i), U_i \in U$, we update them in each iteration as:

$$p(I_j) = \frac{\sum_{E_{ij} \in E} W_{ij}^U \times p(U_i)}{\sum_{E_{ij} \in E} W_{ij}^U}, \tag{1}$$

$$p(U_i) = \frac{\sum_{E_{ij} \in E} W_{ij}^I \times p(I_j)}{\sum_{E_{ij} \in E} W_{ij}^I}, \tag{2}$$

where $W_{ij}^U = \frac{1}{\sum_{I_j \in I} \delta((U_i, I_j) \in E)}$, $W_{ij}^I = \frac{1}{\sum_{U_i \in U} \delta((U_i, I_j) \in E)}$, and $\delta(\cdot)$ is the indicator that gives 1 when the condition is true and 0 otherwise. From the iteration process, it is easy to prove that $p(I_j)$ and $p(U_i)$ will converge after a few iterations.

Fraud Detection. After obtaining the fraud score for each item and the dishonest score for each user, for $I_i \in I$, we label it as a fraud item if $p(I_i) > \zeta$, where ζ is a predefined threshold value. In practice, we can determine a proper ζ through statistically analyzing the fraud score distribution of a small group of benign and fraud users. Note that, it is also possible for us to determine the dishonest users based on their dishonest scores. However, in this paper, we mainly focus on fraud item detection.



(a) T^b's distribution (b) T^f's distribution

Fig. 2. Distribution of T^b and T^f.

3) Time Series based Detection (TSD): In this subsection, we discuss the TSD module, which is designed for detecting fraud items leveraging their time series of incoming traffic. It has been observed in practice that, when a new fraud pattern appears or a new item becomes to a fraud item, the traffic time series, i.e., $T_i = \{V_i^1(t), V_i^2(t), \cdots, V_i^n(t)\}$, of the target item is likely to exhibit differently from the benign items. For example, we pick 1000 benign items with 86777 similar click records, denoted by $I^b = \{I_i | i = 1, 2, \cdots, 1000\}$, and 1000 fraud items with 107768 similar click records, denoted by $I^f = \{I_j | j = 1, 2, \cdots, 1000\}$, from the `Taobao` platform.

Let $T^b = \cup_{I_i \in I^b} T_i$ and $T^f = \cup_{I_j \in I^f} T_j$. If we set $n = 7$ and $t = 1$ day, the distributions of T^b and T^f are shown in Fig. 2. From Fig. 2, we can see that T^b roughly follows one Poisson distribution, while T^f's distribution is more likely the mixture of two Poisson distributions. The reason is intuitive: T^b is generated from the activities of benign users while T^f is generated from the activities of both benign and dishonest users. Based on this fact, it is possible for us to detect fraud items leveraging their traffic time series, especially the traffic time series of a new fraud item.

Following the above idea, we design TSD to model the traffic time series of each item using a *mixture Poisson distribution* (in particular, the mixture of two Poisson distributions) and then derive an *abnormal score* of the traffic time series for each item. Finally, TSD determines the fraud items based on the abnormal scores.

Traffic Time Series Modeling. Now, given an item $I_i \in I$ and its traffic time series $T_i = \{V_i^1(t), V_i^2(t), \cdots, V_i^n(t)\}$, we first make a hypothesis that T_i *follows the mixture of two Poisson distributions* $\mathcal{P}_1$ *and* $\mathcal{P}_2$ *with parameters* $(\lambda_1, \lambda_2, \pi_1, \pi_2)$, *where* λ_1 *and* λ_2 *are the mean values of* $\mathcal{P}_1$ *and* $\mathcal{P}_2$ *respectively,* π_1 *and* π_2 *indicate the mixture ratios of* $\mathcal{P}_1$ *and* $\mathcal{P}_2$ *respectively, and* $\pi_1 + \pi_2 = 1$. Now, for I_i, we define a *latent state set* for its T_i, denoted by $Z_i = \{Z_i^1, Z_i^2, \cdots, Z_i^n\}$, where $Z_i^j \in \{1, 2\}$ is an indicator to indicate that $V_i^j(t)$ is generated from $\mathcal{P}_1$, if $Z_i^j = 1$, or $\mathcal{P}_2$, if $Z_i^j = 2$.

Now, for an item I_i, we use two Poisson distributions $\mathcal{P}_1$ and $\mathcal{P}_2$ to fit the log likelihood function of its traffic time series T_i, which is defined as

$$L(\lambda_1, \lambda_2 | T_i) = \sum_{V_i^j(t) \in T_i} \log \sum_{k=1}^{2} \pi_k p(V_i^j(t) | \lambda_k), \quad (3)$$

where $p(V_i^j(t)|\lambda_k)$ is the probability that $V_i^j(t)$ is generated by $\mathcal{P}_k$ given λ_k. By substituting $p(V_i|\lambda_k)$ with $\frac{\lambda_k^{V_i^j(t)}}{V_i^j(t)!} e^{-\lambda_k}$, the above function is simplified as

$$L(\lambda_1, \lambda_2 | T_i) = \sum_{V_i^j(t) \in T_i} \log \sum_{k=1}^{2} \pi_k \frac{\lambda_k^{V_i^j(t)}}{V_i^j(t)!} e^{-\lambda_k}. \quad (4)$$

Our goal here is to find such $(\lambda_1^\star, \lambda_2^\star, \pi_1^\star, \pi_2^\star)$ that could maximize the above log likelihood function L.

Since it is costly to search the whole solution space to find $(\lambda_1^\star, \lambda_2^\star, \pi_1^\star, \pi_2^\star)$, we employ the Expectation Maximization (EM) framework [10] to learn $(\lambda_1^\star, \lambda_2^\star, \pi_1^\star, \pi_2^\star)$ by iteratively constructing and optimizing the lower-bound of $L(\lambda_1, \lambda_2|T_i)$, which is defined as

$$L(\lambda_1, \lambda_2 | T_i) \geqslant \sum_{V_i^j(t) \in T_i} \sum_{k=1}^{2} Q_i(Z_i^j = k) \log \frac{\pi_k \frac{\lambda_k^{V_i^j(t)}}{V_i^j(t)!} e^{-\lambda_k}}{Q_i(Z_i^j = k)}, \quad (5)$$

where $Q_i(Z_i^j = k)$ is the posterior probability of $Z_i^j = k$. According to the above lower bound, we seek $(\lambda_1^\star, \lambda_2^\star, \pi_1^\star, \pi_2^\star)$ as follows: (1) initializing λ_1 and λ_2 as two positive values (e.g., some values in $[1, 100]$), denoted by λ_1^0 and λ_2^0 respectively, and initializing π_1 and π_2 as some random values in $[0, 1]$, denoted by π_1^0 and π_2^0 respectively, such that $\pi_1 + \pi_2 = 1$; (2) Then, we iteratively optimize the lower bound of $L(\lambda_1, \lambda_2|T_i)$ by executing two steps: Expectation step (E-step) and Maximization step (M-Step). Specifically, in the m-th iteration (we use the superscript m to indicate the value of each variable in the m-th iteration): a) *E-step*: For each $V_i^j(t) \in T_i$, update the posterior probablity $Q_i^m(Z_i^j = k)$ as:

$$Q_i^m(Z_i^j = k) = \frac{\pi_k^{m-1} \frac{\lambda_k^{m-1}}{V_i^j(t)!} e^- \lambda_k^{m-1}}{\sum_{k=1}^{2} \pi_k^{m-1} \frac{\lambda_k^{m-1}}{V_i^j(t)!} e^- \lambda_k^{m-1}}. \quad (6)$$

b) *M-step.* Update π_k and λ_k as

$$\pi_k^m = \frac{\sum_{V_i^j(t) \in T_i} Q_i^m(Z_i^j = k)}{n}, \quad (7)$$

$$\lambda_k^m = \frac{\sum_{V_i^j(t) \in T_i} Q_i^m(Z_i^j = k) \times V_i^j(t)}{\sum_{V_i^j(t) \in T_i} Q_i^m(Z_i^j = k)}. \quad (8)$$

We repeat the iteration until $(\lambda_1, \lambda_2, \pi_1, \pi_2)$ converges[6] and we take the final result as $(\lambda_1^\star, \lambda_2^\star, \pi_1^\star, \pi_2^\star)$.

Fraud Detection. After finding $(\lambda_1^\star, \lambda_2^\star, \pi_1^\star, \pi_2^\star)$, we define the *abnormal score* of I_i as $|\frac{\lambda_2^\star - \lambda_1^\star}{\max\{\lambda_1^\star, \lambda_2^\star\}}|$. Then, if the abnormal score of an item is greater than θ, we label it as a fraud item. Again, in practice, we can determine a proper θ through statistically analyzing the abnormal score distribution of benign and fraud users.

B. Implementation

Through collaborating with `Taobao`, we implement ATF on the Open Data Processing Service (ODPS)[7] platform provided by `Alibaba`. Specifically, we implement the preprocessor in Python using ODPS SQL, implement the GBD module in Java using ODPS SDK, and implement the TSD module in Java. In ATF, the data are stored in a distributed file system and MapReduce is supported for computing and processing.

III. Experimental Evaluation

In this section, we evaluate ATF using labeled datasets. We will deploy ATF on the real `Taobao` platform and perform further analysis in the next section.

A. Datasets and Settings

To validate the performance of ATF, we obtained two real large-scale labeled e-commerce datasets from `Alibaba`:D_1 and D_2, as shown in Table I. Both two datasets are generated in 2016. Based on D_1 and D_2, we can construct two user-item bipartite graphs G_1 and G_2, respectively. According to our analysis, G_1 and G_2 are connected. Let the time slot be 1 day. Then, we can also construct the time series for each item in D_1 and D_2. Note that, these two datasets are used for examining the performance of ATF, and any derived information (e.g.,

[6]The convergence proof of this EM framework can be found at [10].

[7]https://www.aliyun.com/product/odps/

TABLE I
DATASETS. FI = FRAUD ITEMS, BI = BENIGN ITEMS, DU = DISHONEST USERS, AND BU = BENIGN USERS.

Name	#FI	#BI	#DU	#BU	#items	#users	#clicks
D_1	5,500	2,493,625	127,455	46,886,056	2,499,125	47,013,511	156,667,300
D_2	1,100	502,193	29,933	13,646,663	503,293	13,676,596	23,233,154

the fraud item-benign item ratio) does not represent the true scenario of Alibaba.

In our experiment, leveraging map reduce techniques, we run ATF using 228 virtual machines. Each virtual machine is equipped with 2 CPUs and 8 GB memory.

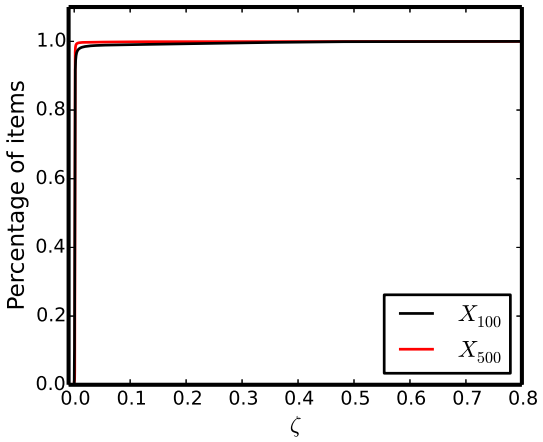


Fig. 3. Cumulative distribution of items over the fraud scores.

TABLE II
PERFORMANCE OF ATF.

	Precision	Recall	F-score
D_1	0.9764	0.9785	0.9774
D_2	0.9749	0.9872	0.9810

B. Performance

Now, we test ATF on D_1 and D_2. We set the dishonest users as seeds, $\zeta = 0.04$, $\theta = 0.9$ and $t = 1$ day. The precision, recall and F-score of ATF on the two datasets are shown in Table II. From Table II, we can see that ATF has high precision, recall and F-score on both datasets, which suggest that ATF is very effective.

Speed and Scalability. Now, we study the speed and scalability of ATF's two detecting modules: GBD and TSD. For this purpose, we randomly sample sub-datasets with 0.3, 0.6, 1.1 and 2.5 million of items from D_1, and then measure the runtime of ATF on those sub-datasets. Fig. 4 plots the runtime v.s. the number of items, showing that both TSD and BSD scale linearly with respect to the dataset size and are very fast. Since GBD and TSD are paralleled on the ODPS platform, GBD finishes the running within 5 minutes and TSD finishes running with 28 minutes even if the dataset has more than 2 million items. This suggests that ATF is scalable in practice and can be applied on large-scale e-commerce platforms.

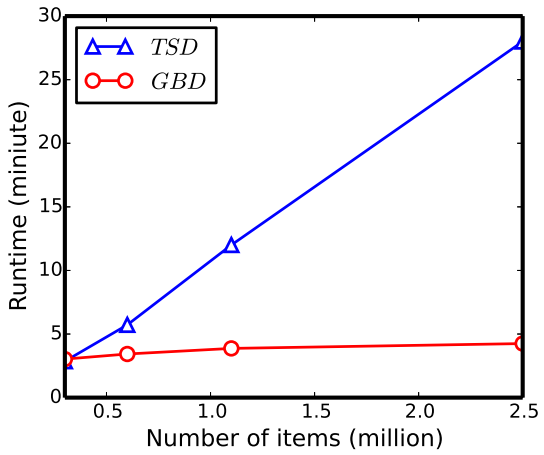


Fig. 4. Runtime v.s. number of items.

IV. ONLINE APPLICATION AND ANALYSIS

In this section, we deploy ATF on Alibaba's real e-commerce platform Taobao to evaluate the performance of ATF in practice. Specifically, we deploy ATF on Taobao to detect the fraud items in eight categories: *men's clothing, women's clothing, men's shoes, women's shoes*, *computer & office, phone & accessories, food & grocery* and *sports & outdoors*, which has over $O(100$ million$)$-scale of active users and $O($billion$)$-scale of items. In the evaluation, we set $\zeta = 0.04$ and $\theta = 0.9$. For GBD, we use a group of dishonest users provided by Alibaba as seeds. For TSD, we set the time slot $t = 1$ day and the time window as $T = 30$ days.

After running ATF, it detects fraud items from each of the eight categories. Then, we report the results to Alibaba. Through the analysis of domain experts, Alibaba confirms that 98.16% of the detected results of ATF are fraud items, which suggests that ATF is very effective in practice.

TABLE III
RUNNING RESULTS OF ATF ON TAOBAO

Category	GBD (%)	TSD (%)	overlap (%)
men's clothing	92.26%	9.99%	2.26%
women's clothing	74.84%	27.49%	2.33%
men's shoes	92.6%	9.41%	2.01%
women's shoes	77.94%	25.12%	3.06%
computer & office	97.66%	3.12%	0.7%
phone & accessories	88.91%	12.68%	1.60%
food & grocery	77.57%	23.56%	1.14%
sports & outdoors	89.08%	12.68%	1.76%

We then evaluate the percentage of fraud items detected by the GBD module and the TSD module, respectively. The results are shown in Table III. From Table III, we can see that:

(1) most of the fraud items are detected by the GBD module. For instance, among the fraud items in the *women's clothing* category, 77.57% of them are reported by GBD and 23.56% of them are reported by TSD; (2) there is an overlap between the fraud items detected by GBD and TSD. For instance, within the fraud items in the *women's clothing* category, 2.33% fraud items are reported by both GBD and TSD; and (3) through the collaboration with the domain experts from Alibaba, we take a further look at the results detected by GBD and TSD. Interestingly, we find that most of the fraud items reported by GBD follow existing promotion patterns, while most of the fraud items reported by TSD are either new fraud items or are promoted using some new promotion patterns. This is mainly because that GBD is bootstrapped by confirmed dishonest users and it tends to find fraud items that are structurally and behaviorally similar to known fraud items, while TSD detects fraud items based on abnormal traffic time series, which is more suitable for detecting new fraud items and fraud items following new promotion patterns.

By inspecting the executing log file of ATF, we find that the running time is 18.11 minutes on average for the GBD module over $O(100$ million$)$-scale of active users and $O($billion$)$-scale of items. The running time of TSD is less than 0.05 seconds on each item.

V. Related Work

Graph Based Fraud Detection. Recently, many fraud detection work has focused on using graphs for spotting frauds [3], [4], [5], [11], [6]. Based on the characteristics of crowd frauds, Tian *et al.* proposed an effective crowd fraud detection method for search engine advertising [3]. Li *et al.* automatically detected the search engine click frauds based on bipartite graph propagation [4]. Van *et al.* leveraged the propagation algorithm to study the influence of network information for tax fraud detection [5]. Tseng *et al.* proposed a graph-based fraudulent phone call detection method to automatically annotate malicious phone numbers with fraud tags [6]. They employed a weighted HITS algorithm to learn the trust value of a phone number and built two bipartite graphs to represent the telecommunication behavior of users. Hu *et al.* propose a bipartite graph-based propagation method for online mobile advertising fraud detection [11].

Mixture Model based Fraud Detection. Mixture models leverage the mixture of parametric statistical distributions to model the normal instances and abnormal instances[12], [14], [13]. Abraham *et al.* assumed that the normal and anomalies are both generated from Gaussian distributions, while the anomalies have a larger variance [12]. Byers *et al.* used a Poisson mixture model to characterize the normal data and then detect anomalies that does not belong to any of the learnt models [14].

Remark. Different from most of the existing fraud detection techniques, ATF's application domain is the online e-commerce marketplace, and it aims to detect various frauds of malicious promotions. More importantly, ATF has been applied on one of the world's largest online e-commerce platforms, Taobao.

VI. Conclusion

In this paper, we study the online e-commerce fraud detection problem. First, we develop an efficient and scalable large-scale e-commerce fraud detection system, named ATF and implement it on the ODPS platform provided by Alibaba. Second, we evaluate ATF leveraging two real large-scale e-commerce datasets. The results indicate that ATF can achieve both high precision and high recall. Third, we further deploy ATF on the Taobao platform of Alibaba, which is one of the world's largest e-commerce platforms. Through evaluation, we demonstrate that ATF is also very effective and scalable in practical scenarios. Our research in this paper is expected to shed light on defending against frauds for large-scale online e-commerce platforms.

Acknowledgment

This work was partly supported by NSFC under No. 61772466 and No. 61472359, the Provincial Key Research and Development Program of Zhejiang, China under No. 2017C01055, the Fundamental Research Funds for the Central Universities, the Alibaba-Zhejiang University Joint Institute of Frontier Technologies, the CCF-NSFOCUS Research Fund under No. CCF-NSFOCUS2017011, and the CCF-Venustech Research Fund under No. CCF-VenustechRP2017009.

References

[1] C.-H. Park and Y.-G. Kim, "Identifying key factors affecting consumer purchase behavior in an online shopping context," *International Journal of Retail & Distribution Management*, 2003.

[2] J. Lim, https://www.forbes.com/sites/jlim/2015/04/11/jd-com-brushing-fake-orders-to-inflate-sales.

[3] T. Tian, J. Zhu, F. Xia, X. Zhuang, and T. Zhang, "Crowd fraud detection in internet advertising," in *ACM WWW 2015*.

[4] X. Li, M. Zhang, Y. Liu, S. Ma, Y. Jin, and L. Ru, "Search engine click spam detection based on bipartite graph propagation," in *ACM WWW 2014*.

[5] V. Van Vlasselaer, T. Eliassi-Rad, L. Akoglu, M. Snoeck, and B. Baesens, "Gotcha! network-based fraud detection for social security fraud," *Management Science*, 2016.

[6] V. S. Tseng, J.-C. Ying, C.-W. Huang, Y. Kao, and K.-T. Chen, "Fraudetector: A graph-mining-based framework for fraudulent phone call detection," in *ACM SIGKDD 2015*.

[7] D. Proserpio, S. Goldberg, and F. McSherry, "Calibrating data to sensitivity in private data analysis," in *PVLDB 2014*.

[8] C. Liu, P. Gao, M. Wright, and P. Mittal, "Exploiting temporal dynamics in sybil defenses," in *ACM CCS 2015*.

[9] J. M. Kleinberg, "Authoritative sources in a hyperlinked environment," *Journal of the ACM*, 1999.

[10] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the royal statistical society. Series B (methodological)*, 1977.

[11] J. Hu, J. Liang, and S. Dong, "ibgp: A bipartite graph propagation approach for mobile advertising fraud detection," *Mobile Information Systems*, 2017.

[12] B. Abraham and G. E. Box, "Bayesian analysis of some outlier problems in time series," *Biometrika*, 1979.

[13] M. Bahrololum and M. Khaleghi, "Anomaly intrusion detection system using hierarchical gaussian mixture model," *International journal of computer science and network security*, 2008.

[14] S. Byers and A. E. Raftery, "Nearest-neighbor clutter removal for estimating features in spatial point processes," *Journal of the American Statistical Association*, 1998.