Fraud Regulating Policy for E-Commerce via Constrained Contextual Bandits

Zehong Hu
Alibaba Group
Hangzhou, China
zehong.hzh@alibaba-inc.com

Zhen Wang
Alibaba Group
Hangzhou, China
jones.wz@alibaba-inc.com

Zhao Li
Alibaba Group
Hangzhou, China
lizhao.lz@alibaba-inc.com

Shichang Hu
Alibaba Group
Hangzhou, China
shichang.hsc@alibaba-inc.com

Shasha Ruan
Alibaba Group
Hangzhou, China
shasha.rss@alibaba-inc.com

Jie Zhang
Nanyang Technological University
Singapore
zhangj@ntu.edu.sg

ABSTRACT

Fraud sellers in e-commerce often promote themselves via fake visits or purchases to increase sales, jeopardizing the business environment of the platform. How to regulate the exposure of these sellers to buyers without affecting normal online business remains a challenging problem, since blocking them entirely without discrimination may kill the normal transactions and could potentially decrease the total transactions of the platform. To address this problem, we introduce a regulating valve which blocks fraud sellers with a certain probability. To learn the optimal blocking policy, we model the regulating valve as a contextual bandit problem with a constraint on the total transaction decline. Since existing bandit algorithms are unable to incorporate the transaction constraint, we propose a novel bandit algorithm, which decides the policy based on a set of neural networks and iteratively updates the neural networks with online observations and the constraint. Experiments on synthetic data and one of the largest e-commerce platforms in the world both show that our algorithm effectively and efficiently outperforms existing bandit algorithms by a large margin.

CCS CONCEPTS

• **Applied computing** → **Electronic commerce**; **Online shopping**;

KEYWORDS

E-Commerce; Fraud Sellers; Buyer Impressions; Constrained Contextual Bandits

ACM Reference Format:
Zehong Hu, Zhen Wang, Zhao Li, Shichang Hu, Shasha Ruan, and Jie Zhang. 2019. Fraud Regulating Policy for E-Commerce via Constrained Contextual Bandits. In *Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), Montreal, Canada, May 13–17, 2019,* IFAAMAS, 9 pages.

1 INTRODUCTION

The main function of e-commerce platforms is to guide buyer impressions to sellers, where a buyer impression means the exposure to one buyer. Currently, buyer impressions in e-commerce are allocated through a highly complex ranking system which selects top-k sellers based on their quality scores and how good they can match buyers' queries and preferences [17, 20]. In reality, more buyer impressions often give rise to more sales, and the ranking system often prefers popular sellers as they can potentially bring more sales for the platform. Thus, with the goal of increasing sales, some sellers fraudulently boost their popularity by, for example, hiring a group of human labors to visit their shops frequently or buy items from their shops and give good feedbacks [19]. In this way, these fraud sellers can gain unfair advantages over others, which jeopardizes the business environment of the platform. To keep the market fair, the platform must take away some buyer impressions from these fraud sellers. On the other hand, these fraud sellers also contribute a certain amount of normal transactions to the platform. In addition, the fraud detection techniques used to identify fraud sellers are possible to make false positive decisions. Hence, reducing the buyer impressions of fraud sellers too much may mistakenly kill some normal transactions, which will cause the total transactions of the platform to decline. With all these factors considered, it is a long-standing challenge for e-commerce platforms to find a proper policy to regulate the buyer impressions of fraud sellers.

To regulate the buyer impression of fraud sellers, previous researchers propose a few fraud-combating mechanisms which consider sellers' fraudulent behaviors when computing their ranking scores [7, 8, 30]. However, the applicability of these mechanisms is questionable because they may not be easily incorporated into the highly complex ranking system used in existing e-commerce platforms [17, 20]. To be more specific, these mechanisms are designed to re-construct the ranking system on seller side. They yet are not able to handle the queries and preferences of buyers. However, the most fundamental function of e-commerce platforms is to satisfy buyers' queries and preferences in the best way, which is exactly the objective of the existing ranking systems. It is also worth mentioning that current ranking systems regulate buyer impressions received by fraud sellers via manually setting a negative bias on their ranking scores. Nevertheless, how the bias will affect fraud sellers is unknown because the ranking scores will change greatly

as buyers' queries and preferences as well as the historical transaction data of the products provided by fraud sellers. In this paper, instead of redesigning the ranking system, we propose to regulate the buyer impressions of fraud sellers by adding a regulating valve into the existing ranking system, which blocks fraud sellers with a certain probability. Moreover, since how the regulating valve affects the normal transactions is unknown, we first model the regulating valve via contextual bandits with an extra constraint on the transaction decline. Then, we propose a novel bandit algorithm which, based on the online observations of buyer impressions and transactions, learns the optimal fraud regulating policy to minimize the buyer impressions of fraud sellers and meanwhile keep the transaction constraint unviolated.

To the best of our knowledge, our work is the first attempt to regulate the buyer impressions of fraud sellers in e-commerce with constrained contextual bandits. Our contribution consists of two aspects. First, we propose a novel way of buyer impression regulating by adding a regulating valve into the existing ranking system of e-commerce platforms. When a buyer types in a query, it stops fraud sellers from entering the ranking system with a certain probability. Since buyers and queries come very fast on e-commerce platforms, the high-speed switch between blocking and unblocking can accurately cut away a certain ratio of buyer impressions from fraud sellers. Then, considering different buyers may have different levels of preferences about fraud sellers, we should learn a policy to dynamically adjust the blocking probability so that the mistaken kill of normal transactions can be avoided as many as possible. Thus, we model our regulating valve as a constrained contextual bandit problem, where the reward is the ratio of buyer impressions received by non-fraud sellers and the constraint is the transaction decline. Since the existing contextual bandit algorithms either cannot incorporate the constraint or need strong assumptions that are not satisfied in our problem, we propose a novel constrained contextual bandit algorithm which uses a set of neural networks to set up a policy pool and randomly select one for decision-making at each step. These neural networks are independently updated by sampling historical observations. To ensure the constraint to be satisfied, our algorithm updates neural networks by formulating a local optimization problem via linearizing both buyer impressions and transactions. We conduct extensive experiments on both synthetic data and one of the largest e-commerce platforms in the world, which show that our algorithm significantly outperforms existing contextual bandit algorithms. Experiments on the real-world e-commerce platform also show that our algorithm considerably reduces the buyer impressions of fraud sellers, yet with only slight transaction decline.

2 RELATED WORK

In this section, we review existing ranking systems, fraud-combating mechanisms, and contextual bandit algorithms.

2.1 Ranking Systems

When buyers type in a query, e-commerce platforms need to rank the items of different sellers and display the top-k items in a page. To satisfy buyers' preferences and queries in the best way, e-commerce platforms often learn the ranking function from buyers' operations on the page, termed as learning to rank (LTR). Early learning to rank systems are offline [6, 16]. Later on, online learning and reinforcement learning techniques are introduced to improve the ranking in an online fashion [12, 20, 32]. Notwithstanding the success of these systems, their ability to regulate buyer impressions of fraud sellers is poor. One commonly used method is to manually set a negative bias on the ranking scores of fraud sellers. However, the effects of the bias on fraud sellers are actually unknown. For some queries, alternative sellers are few, and the fraud sellers will remain in the top-k no matter what bias is set. For others, when there are many alternative sellers, a small bias usually means 100% reduction of buyer impressions. In addition, for different items of the fraud sellers, the ranking scores often change greatly. Even for one item, buyers' operations (e.g. buy, click or add to the shopping cart) might also cause the ranking score to change significantly. In this case, selecting a proper bias is usually very difficult. Thus, in this paper, we propose a novel way to regulate the buyer impressions of fraud sellers. We add a regulating valve at the entrance of the ranking system which blocks fraud sellers from entering the ranking system with a certain probability. Through the high-speed switch between blocking and unblocking, we can always precisely cut away a certain ratio of buyer impressions from fraud sellers, no matter how the ranking score changes in reality.

2.2 Fraud-Combating Mechanisms

Recently, to combat fraud sellers, Cai *et al.* [7–9] and Zhao *et al.* [30] leverage emerging reinforcement mechanism design techniques [26] to build new ranking systems, termed as fraud-combating mechanisms. However, the applicability of these mechanisms is questionable because they do not consider how to satisfy buyers' queries and preferences, which is the fundamental objective of ranking systems. For example, the ranking in the mechanism designed by Zhao *et al.* [30] is decided only based on sellers' six features, including prices, the number of clicks and so on. In this case, it is difficult to apply these fraud-combating mechanisms mechanism in real e-commerce platforms. Thus, in this paper, instead of redesigning new ranking systems, we add a regulating valve at the entrance of the existing ranking systems to block fraud sellers with a certain probability. In this way, we can equip existing ranking systems with the ability to regulate the buyer impressions of fraud sellers, yet without making any inside changes.

2.3 Contextual Bandits

In contextual bandits, an agent repeatedly interacts with the environment [21]. At each step, it firstly observes a context vector x. Then, it chooses an arm a and obtains a reward from the environment. The reward function $r(x, a)$ maps contexts and arms to real-valued rewards. The agent expects maximum rewards but does not know the reward function. To balance the exploration for the reward function and the greed for maximum rewards, two groups of algorithms have been proposed. One group uses upper confidence bounds of reward function estimations to decide the optimal arm, including Lin-UCB [1, 10, 15], GLM-UCB [11] and so on [14, 23, 29]. Another group decides the optimal arm by sampling an estimate of the reward function from the posterior distribution of rewards [4]. Since calculating posterior distributions is challenging,

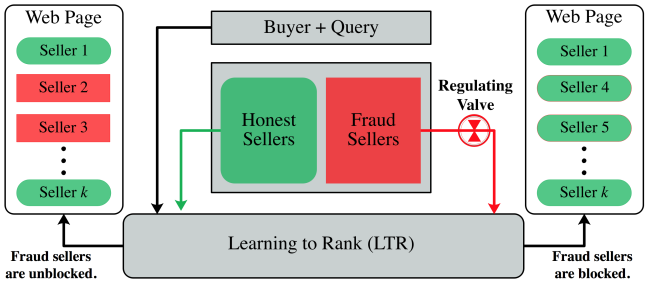


Figure 1: Schematic diagram of fraud regulating

Lu and Van Roy [18] propose ensemble sampling which calculates a set of reward function estimations instead.

The aforementioned contextual bandit algorithms are all value-based. Due to the constraint on transaction decline, they need to learn two models when applied to our problem: one for rewards and another for transactions. At each step, to approximate the optimal arm selection policy, we still need to solve a programming problem based on context samples and the estimates of the two models. However, the two models and the approximate programming all generate certain levels of errors. The combination of these errors will make it difficult for bandit algorithms to find the optimal policy. In this paper, we propose, to the best of our knowledge, the first policy-based constrained contextual bandit algorithm. It uses neural networks to parameterize the arm selection policies and directly learns the optimal policy based on reward and transaction observations. In this way, we can avoid approximately solving the optimal policy based on inaccurate estimates.

Note that, there are some existing studies on constrained contextual bandits [3, 25, 27]. However, these algorithms are unable to solve our problem because they need strong assumptions that are not satisfied in our problem. For example, Wu et al. [27] focus on the contextual bandit problem which generates a certain cost at each step and the total budget of costs is limited. They assume the number of contexts is finite and the distribution of contexts is known. Under these two assumptions, they construct an optimization problem over contexts at each step and decide the optimal policy by solving this optimization problem. Nevertheless, in our problem, contexts are high-dimensional continuous vectors and the distribution is unknown, making their algorithm unable to be applied. Agrawal et al. [3] and Sun et al. [25] focus on the constrained contextual bandit problem where a set of expert policies is known in advance. Based on the online observations, their algorithms can learn the optimal weighted combination of expert policies. However, in our fraud regulating problem, it is not easy to find a feasible policy, not to mention a set of expert policies.

3 REGULATING VALVE

The schematic diagram of our fraud regulating system is shown in Figure 1. Traditional LTR systems focus on satisfying buyers by carefully selecting the top-k sellers. A fundamental assumption of their learning process is that the data collected to describe buyers and sellers are truthfully provided. Developing LTR systems based on this assumption makes them unable to defend the fraudulent behaviors of sellers. Thus, in this paper, we add a regulating valve at the entrance of the LTR system to block fraud sellers with a certain probability. When the regulating valve blocks fraud sellers, only honest sellers can enter ranking and be displayed to buyers in the web page. When fraud sellers are unblocked, both honest and fraud sellers can be displayed. Since there are usually thousands of buyers and queries coming to the LTR system of e-commerce platforms in one second, the switch between blocking and unblocking is very fast, which offers us the ability to accurately take away a certain ratio of buyer impressions from fraud sellers. Furthermore, to avoid the mistaken kill of normal transactions, the blocking probability should take buyers' preferences and queries into consideration. However, how the blocking probability affects the normal transactions is unknown and may change greatly as time. Thus, we need to develop an intelligent algorithm to automatically learn the optimal fraud regulating policy based on online observations.

To this end, we formulate our fraud regulating system at first and introduce some notations as follows. In e-commerce, buyers with queries come to the platform sequentially. We denote the buyer and query coming at step $t = 1, 2, \ldots, T$ by a feature vector x_t. The list of sellers shown to the buyer is written as a vector $\langle s_t(1), \ldots, s_t(k)\rangle$, where $s_t(i)$ denotes the seller at the i-th slot of the web page. In addition, we denote the set of fraud sellers by S_f. Then, the buyer impressions received by fraud sellers at step t can be calculated as

$$\mathsf{FIM}_t = \sum_{i=1}^{k} 1\left[s_t(i) \in S_f\right] \tag{1}$$

where $1[\cdot]$ is an indicator function. After seeing the web page, the buyer may conduct some operations, for example, buying an item. We write the transactions generated at step t as PT_t. Besides, we use $a_t \in \{0, 1\}$ to denote the state of our regulating valve, where 1 and 0 mean blocking and unblocking, respectively. Note that the fraud detection in e-commerce usually needs several days to update the set of fraud sellers S_f [24, 28] while thousands of buyers come to the platform every second. In this case, we can regard the updating of fraud sellers happens at $t = +\infty$, where the regulating policy considered in this paper should have converged.

The objective of the regulating valve is to reduce the buyer impressions received by fraud sellers. On the other hand, the objective of ranking is to match buyers and sellers in the best way so that the most transactions can be generated. Our regulating valve blocks a part of sellers, which narrows down the choices of ranking and thus inevitably has some negative effects on transactions. From the perspective of e-commerce platforms, any significant decrease of the normal transactions is intolerable. Thus, we can formulate the regulating valve as the following problem

$$\min_{(a_1,\ldots,a_T)} \quad \sum_{t=1}^{T} \mathsf{FIM}_t \tag{2}$$

$$\text{s.t.} \quad \sum_{t=1}^{T} \mathsf{PT}_t - V_0 \geq d \cdot V_0 \tag{3}$$

where d is the lower bound of the transaction change rate and V_0 is the baseline amount of transactions when the regulating valve is not used. When $d < 0$, the right-hand side of Equation 3 denotes the transaction decline compared to the baseline.

Since T is very large, it is difficult to solve the global summation of buyer impressions FIM_t and transactions PT_t over T steps and decide the exact value of V_0. Thus, we divide the two sides of

Equations 2 and 3 by T and rewrite the two equations as

$$\min_{(a_1,\ldots,a_T)} \quad \frac{1}{T}\sum_{t=1}^{T} \mathsf{FIM}_t \tag{4}$$

$$\text{s.t.} \quad \frac{1}{T}\sum_{t=1}^{T} \frac{\mathsf{PT}_t - v_0}{v_0} \geq d \tag{5}$$

where $v_0 = V_0/T$ denotes the average transaction per query. In practice, to decide the value of v_0, we randomly allocate buyers and queries into two channels—one with and one without the regulating valve, which is similar to the widely-adopted A/B test [13]. In the channel without the regulating valve, we maintain a moving window to calculate the latest value of v_0. For Equations 2 and 3, since FIM_t and PT_t are not known, we cannot directly solve it. In the next section, based on contextual bandits, we will develop a novel algorithm which can gradually learn the optimal regulating policy from the online observations of FIM_t and PT_t.

4 CONSTRAINED CONTEXTUAL BANDITS

In this section, we model the regulating valve in Figure 1 as a constrained contextual bandit problem. At each step t, we firstly observe a context vector x_t and need to choose one arm $a_t \in \{0, 1\}$. After choosing the arm, we can observe one reward signal $r_t = (k - \mathsf{FIM}_t)/k$ and one constraint signal $c_t = (\mathsf{PT}_t - v_0)/v_0$, where the reward signal r_t denotes the ratio of buyer impressions received by honest sellers. Then, Equations 4 and 5 are equivalent to a constrained contextual bandit problem of which the objective is to accumulate the maximum rewards and the constraint is to keep the average of c_t not lower than d. Existing contextual bandit algorithms are all value-based. They maintain an estimation about rewards. At each step, they greedily select the optimal arm and update the estimation with new observations. The charm of bandit algorithms originates from the method to select the optimal arm, which can optimally balance the greed for the most rewards and the need to explore uncertain areas of the reward estimation.

When applying value-based bandit algorithms to our problem, we need to maintain two models to estimate r_t and c_t, respectively. Then, based on these estimates and Equations 2 and 3, if we want to apply the idea of Wu et al.'s studies [27], we need to solve a linear programming problem with T variables to get the optimal arm selection policy. The time complexity of linear programming is polynomial to T [5], whereas the online decision of our regulating valve needs to be very fast. In this case, we can only approximate the optimal policy by replacing T with a small value. This approximation will cause arm selection to deviate from the optimal policy, even if r_t and c_t have been accurately estimated. Not to mention both the estimates of r_t and c_t inevitably have certain levels of errors. Thus, we can conclude that existing bandit algorithms are unable to learn the optimal policy in our problem.

In this regard, we propose a novel policy-based bandit algorithm to solve the constrained contextual bandit problem. Instead of relying on the estimates of r and c, we use neural networks to represent the arm selection policy and learns the optimal policy directly from observations. This way of design avoids the long-winded process of value-based bandit algorithms in handling the constraint. By doing so, our algorithm not only reduces learning errors but also boosts computation efficiency. Another benefit of our design is that we do not need any prior knowledge about buyers and the ranking system because neural network is a generally applicable function approximator that can approximate any function and has been successfully used in many applications. In practice, we also cannot provide any prior knowledge because both the ranking system and buyers' behaviors are very complex and change with time.

To design our algorithm, we first need to know the optimal arm selection policy when all statistics about x, r and c are stationary and known. In this paper, we call this optimal policy as the oracle, and it will serve as the objective of our algorithm. To formally describe the oracle, we introduce $\pi(a|x)$ to denote the arm selection policy given context x. Note that $\pi(a|x)$ is a mixed policy which means a distribution over arms. Furthermore, we introduce $\mathbb{E}_\pi[r]$ and $\mathbb{E}_\pi[c]$ to denote the expectations of r and c, given the arm selection policy π and the distribution of x. Considering T in Equations 4 and 5 is very large, we can replace the average operation $T^{-1}\sum_t$ with expectation. Then, the oracle π^* is the solution of the following optimization problem:

$$\pi^* = \arg\max_\pi \mathbb{E}_\pi[r] \qquad \text{s.t. } \mathbb{E}_\pi[c] \geq d \tag{6}$$

where the threshold d is a given constant. Next, we need to answer two questions: 1) how to approach the oracle by updating neural networks with online observations? 2) how to prevent neural networks from getting trapped in the local optimum? To answer the first question, we linearize both r and c at the local level, and then construct a constrained local optimization problem. By deriving the analytical solution of the local optimization problem, we get a new form of neural network updating. To answer the second question, we use a set of neural networks to set up a policy pool and randomly sample one for decision-making at each step. Our idea of random sampling comes from ensemble sampling, the state-of-the-art bandit algorithm. After answering these two questions, we summarize our algorithm in Algorithm 1.

4.1 Policy Network

In this section, we use neural networks to approach the oracle with online observations. The input of the neural network is the context vector x, and the output is the arm selection policy $\pi(a|x)$. We call this neural network as policy network and denote it by $\pi(a|x;\theta)$, where θ denotes the parameters of the neural network. To approach the oracle, we iteratively update the policy network in the neighborhood of θ, following the same idea as gradient descend, the most popular method to train neural networks [31]. However, the simple gradient cannot incorporate the constraint in our problem. Thus, we need a new form of neural network updating. To this end, we first study the connection of two neighboring polices and get:

Theorem 4.1. *For any stochastic variable $f(a)$, any two neighboring policies $\pi(a|x;\theta_1)$ and $\pi(a|x;\theta_2)$ satisfy*

$$\mathbb{E}_{\pi(\theta_2)}[f(a)] - \mathbb{E}_{\pi(\theta_1)}[f(a)] \approx g_f^{\mathrm{T}}(\theta_1)\cdot(\theta_2 - \theta_1) \tag{7}$$

$$g_f(\theta_1) = \mathbb{E}_q\left[\frac{\nabla_\theta \pi(a|x;\theta_1)}{q(a|x)} f(a)\right] \tag{8}$$

where $q(a|x)$ can be any policy.

Proof. According to importance sampling [22], we can calculate the expectation of $f(a)$ as

$$\mathbb{E}_{\pi(\theta_1)}[f(a)] = \mathbb{E}_q\left[q^{-1}(a|x)\cdot\pi(a|x;\theta_1)f(a)\right]. \tag{9}$$

Then, we can calculate the expectation difference as

$$\mathbb{E}_{\pi(\theta_2)}[f(a)] - \mathbb{E}_{\pi(\theta_1)}[f(a)] = \mathbb{E}_q\left[\frac{\pi(a|x,\theta_2) - \pi(a|x,\theta_1)}{q(a|x)} f(a)\right]. \quad (10)$$

Considering $\pi(\theta_1)$ and $\pi(\theta_2)$ are two neighboring policies, we can conduct linear approximation as

$$\pi(a|x,\theta_2) - \pi(a|x,\theta_1) \approx [\nabla_\theta \pi(a|x;\theta_1)]^{\mathrm{T}} \cdot (\theta_2 - \theta_1) \quad (11)$$

which concludes Theorem 4.1. □

Next, let the parameter of the policy network be θ_i, where the subscript i is the counter of neural network updating. Based on Theorem 4.1, we can linearize the expectation of rewards and transactions in Equation 6 as follows

$$\begin{aligned} \mathbb{E}_{\pi(\theta)}[r] &= \mathbb{E}_{\pi(\theta_i)}[r] + g_r^{\mathrm{T}}(\theta_i) \cdot (\theta - \theta_i), \\ \mathbb{E}_{\pi(\theta)}[c] &= \mathbb{E}_{\pi(\theta_i)}[c] + g_c^{\mathrm{T}}(\theta_i) \cdot (\theta - \theta_i) \end{aligned} \quad (12)$$

where g_r and g_c can be obtained by replacing f in Equation 8 with r and c, respectively. Suppose we follow policy $q(a|x)$ and get N observations, (r_{ij}, c_{ij}), where $j = 1, \ldots, N$. Then, we can approximate the expectation in Equation 12 with the average over N observations and set up the following updating to approach the oracle:

$$\theta_{i+1} = \arg\max_\theta \bar{g}_r^{\mathrm{T}}(\theta_i) \cdot (\theta - \theta_i) \quad (13)$$

$$\text{s.t.} \quad \bar{w} - \bar{g}_c^{\mathrm{T}}(\theta_i) \cdot (\theta - \theta_i) \le 0 \ , \ ||\theta - \theta_i||_2 \le \delta \quad (14)$$

where $||\cdot||_2$ is 2-norm and δ is the step size of neural network updating, which decides the size of the neighborhood.

$$\begin{aligned} \bar{g}_r(\theta_i) &= \frac{1}{N}\sum_{j=1}^{N} \frac{\nabla_\theta \pi(a_{ij}|x_{ij};\theta_i)}{q(a_{ij}|x_{ij})} r_{ij}, \\ \bar{g}_c(\theta_i) &= \frac{1}{N}\sum_{j=1}^{N} \frac{\nabla_\theta \pi(a_{ij}|x_{ij};\theta_i)}{q(a_{ij}|x_{ij})} c_{ij}, \\ \bar{w}(\theta_i) &= d - \frac{1}{N}\sum_{j=1}^{N} \frac{\pi(a_{ij}|x_{ij};\theta_i)}{q(a_{ij}|x_{ij})} c_{ij}. \end{aligned} \quad (15)$$

Note that, in the right hand side of Equation 13, we omit the term $\mathbb{E}_{\pi(\theta_i)}[r]$ because it is a constant and has no effect on the solution. Besides, in Equations 13 and 14, the real running arm selection policy $q(a|x)$ can be different from $\pi(a|x;\theta_i)$. This feature is very important because the ensemble sampling layer discussed in the next subsection will make the real running arm selection policy be the random mixture of a set of policy networks rather than the output of any single policy network.

Equations 13 and 14 form a second-order cone programming problem. They provide a new form of neural network updating which can incorporate the constraint in our problem. To solve this programming problem, we define some notations at first:

$$a_1 = \bar{g}_r^{\mathrm{T}}\bar{g}_r \ , \ a_2 = \bar{g}_r^{\mathrm{T}}\bar{g}_c \ , a_3 = \bar{g}_c^{\mathrm{T}}\bar{g}_c \ , \ a_4 = a_1 - a_2^2/a_3 \ ,$$
$$a_5 = a_2\bar{w}/a_3 \ , \ a_6 = a_2/\bar{w} \ , \ \Delta = \delta^2 - \bar{w}^2/a_3 \ .$$

Then, we can get the following theorems:

Theorem 4.2. *If $\Delta > 0$ or $\bar{w} \le 0$, the solution of the optimization problem defined in Equations 13 and 14 satisfies*

$$\theta_{i+1} = \theta_i + \lambda^{-1}(\bar{g}_r + v\bar{g}_c) \quad (16)$$

where $v = \max\{0, a_3^{-1}(\lambda\bar{w} - a_2)\}$, $\lambda = \lambda_1$ if $J_1(\lambda_1) > J_2(\lambda_2)$ and $\Delta > 0$, and $\lambda = \lambda_2$ otherwise.

$$J_1(\lambda) = -\frac{a_4\lambda^{-1} + \Delta\lambda}{2} - a_5 \ , \ J_2(\lambda) = -\frac{a_1\lambda^{-1} + \delta\lambda}{2} \ .$$

	$\bar{w} > 0$	$\bar{w} < 0$	$\bar{w} = 0$	
			$a_2 \ge 0$	$a_2 < 0$
$\lambda_1 =$	$h(a_6, a_7)$	$h(0, l(a_6, a_7))$	*Null*	a_7
$\lambda_2 =$	$h(0, l(a_6, a_8))$	$h(a_6, a_8)$	a_8	*Null*
$h(x,y) = \max(x,y)$, $l(x,y) = \min(x,y)$ $a_7 = \sqrt{a_4/\Delta}$, $a_8 = \sqrt{a1/\delta}$, *Null= Not Exist*				

The proof of Theorem 4.2 relies on Slater's condition [5] and the strong duality of the second-order cone programming in Equations 13 and 14, which is similar to Theorem 2 in the appendix of [2]. Due to the space limitation, we skip the detailed proof here.

Theorem 4.3. *If $\Delta \le 0$ and $\bar{w} > 0$, when $||\theta - \theta_i||_2 \le \delta$,*

$$C(\theta) \ge 0 \ , \ \arg\min C(\theta) = \theta_i + \sqrt{\delta^2/a_3} \cdot \bar{g}_c \quad (17)$$

where $C(\theta) = \bar{w} - \bar{g}_c^{\mathrm{T}} \cdot (\theta - \theta_i)$ is the left hand side of the first constraint in Equation 14, and $C(\theta) \ge 0$ means the second-order cone programming problem in Equations 13 and 14 has no or only one feasible solution at the boundary $C(\theta) = 0$.

We can prove Theorem 4.3 by using $\theta = \theta_i$ to verify Slater's condition at first and then constructing a Lagrange function similar to the proof of Theorem 4.2. Due to the space limitation, we skip the details here. From Theorem 4.3, we can know that, when $\bar{w} > 0$ and $\Delta \le 0$, Equations 13 and 14 have no or only one feasible solution. In this case, we should greedily alleviate the constraint violation by using the following updating equation:

$$\theta_{i+1} = \theta_i + \rho \cdot \sqrt{\delta^2/a_3} \cdot \bar{g}_c \quad (18)$$

where the risk aversion parameter $\rho \ge 1.0$ ensures our algorithm to put higher priority on satisfying the constraint. To summarize, we update our policy network by Equation 16 when $\Delta > 0$ or $\bar{w} \le 0$ and Equation 18 otherwise.

4.2 Ensemble Sampling

Our policy network approaches the oracle by iteratively solving the local optimum. A drawback of this method is that it is not good at exploring uncertain policies and may get trapped in the local optimum. To solve this problem, we introduce ensemble sampling on top of the policy network. The main idea of ensemble sampling is to use a set of estimates to replace the posterior distribution required by the classic Thompson sampling [18]. Thus, in our algorithm, we maintain n policy networks and randomly sample one for decision-making at each step. In this case, the real arm selection policy in our algorithm becomes a random mixture of the outputs of the n policy networks. We denote the real arm selection policy by $q(a|x_t)$, which corresponds to the denominator of Equation 15.

To summarize, we formally present our algorithm, ensemble sampling with constrained policy networks (ES-CPN), in Algorithm 1. It firstly initializes n policy networks. Then, at each step t, it observes the context x_t, samples one out of the n policies, and observes reward signal r_t as well as constraint signal c_t (lines 4-7). Besides, our algorithm updates the n policy networks by sampling the reply buffer every N steps (lines 8-9). For the parameters in our algorithm,

Algorithm 1: Ensemble Sampling with Constrained Policy Networks (ES-CPN)

Initialize n policy networks $\pi(a|x;\theta_0^1), \pi(a|x;\theta_0^2), \ldots, \pi(a|x;\theta_0^n)$ *randomly as well as the reply buffer;*
for $i = 1, 2, 3 \ldots$ **do**
  for $t = (i-1)\cdot N + 1$ **to** $i \cdot N$ **do**
    Observe the context vector x_t*, and compute n policies as* $\pi(a|x_t;\theta_i^1), \pi(a|x_t;\theta_i^2), \ldots, \pi(a|x_t;\theta_i^n)$*;*
    Uniformly sample one policy, denoted by $\pi(a|x_t;\theta_i^*)$*, and choose the arm* $a_t \sim q(a|x_t) = \pi(a|x_t;\theta_i^*)$*;*
    Observe r_t *and* c_t*, and store* $[x_t, a_t, r_t, c_t, q(a_t|x_t)]$ *into the reply buffer;*
  for $l = 1$ **to** n **do**
    Sample N_{tr} *observations from the latest* N_{me} *observations in the reply buffer to calculate* $\bar{g}_r(\theta_i^l)$*,* $\bar{g}_c(\theta_i^l)$ *and* $\bar{w}(\theta_i^l)$*;*
    Update policy network $\pi(a|x;\theta_i^l)$ *by Equation 16 when* $\Delta(\theta_i^l) > 0$ *or* $\bar{w}(\theta_i^l) \le 0$ *and Equation 18 otherwise;*

N decides the training frequency of policy networks. It cannot be very large; otherwise, the convergence of policy networks will be very slow. n decides the number of policy networks. We should keep it as large as possible so as to improve the exploration of our algorithm. Actually, the n policy networks can be computed in parallel, which is favorable for practical usage. N_{tr} and N_{me} decide the size of the training batch and memory, respectively. The larger they are, the better. However, a large N_{tr} may lead to low computation efficiency, and a large N_{me} may cause our algorithm to respond to environment changes very slowly. Note that, even though our regulating valve only has two arms, our algorithm is generally applicable for constrained contextual bandit problems with an arbitrary number of arms.

5 EXPERIMENTS

In this section, we first use synthetic data to verify the advantages of our algorithm, *ES-CPN*, over existing bandit algorithm. Then, we show the effectiveness of our algorithm in regulating buyer impressions by conducting experiments on real-world data. The benchmarking algorithms include:

- **Lin-UCB** [15] assumes the expectation of rewards satisfies $\mathbb{E}[r|x,a] = x^{\mathrm{T}} \cdot \theta_r(a)$, where θ_a means the parameters corresponding to arm a. At each step t, Lin-UCB computes the upper confidence bounds (UCBs) of the reward estimates as
$$\mathrm{ucb}(x_t, a) = x_t^{\mathrm{T}} \cdot \tilde{\theta}_r(a) + \alpha\sqrt{x_t^{\mathrm{T}} A_a^{-1} x_t} \tag{19}$$
where $\tilde{\theta}_r$ is the estimates of θ_r and A_a^{-1} is the corresponding covariance matrix. α is a constant. Lin-UCB optimistically selects the arm with the maximum upper confidence bound.
- **GLM-UCB** [11] is an extension of Lin-UCB and also selects the arm with the maximum upper confidence bound at each step. For Bernoulli rewards, it assumes the reward satisfies the logistic regression model and computes the upper confidence bound as
$$\mathrm{ucb}(x_t, a) = \mu(x_t^{\mathrm{T}} \cdot \tilde{\theta}_r(a)) + \rho(t)\sqrt{x_t^{\mathrm{T}} A_a^{-1} x_t} \tag{20}$$
where $\mu(x) = \exp(x)/[1+\exp(x)]$ and $\rho(t)$ is a increasing function of t.
- *Ensemble Sampling* (ES) [18] maintains a set of neural networks to estimate rewards and randomly samples one to decide the optimal arm at each step.

All the above contextual bandit algorithms learn a model to estimate rewards at first and then decide the optimal arm based on their estimates. When applying them to our problem, we need to build a separate model for transactions. In our experiments, when testing each algorithm, we use the same type of models for transactions as for rewards. Besides, for contextual bandits with constraints, the simple greedy policy is no longer applicable. Thus, we follow the idea of Wu et al.'s studies ([2015]) and build the following linear programming problem to approximate the oracle at each step t:[1]

$$\begin{aligned} \pi_t^* = &\arg\max_\pi \sum_{j=1}^{b}\sum_a \pi(a|x_{tj}) e_r(a, x_{tj}) \\ &\text{s.t.} \quad \sum_{j=1}^{b}\sum_a \pi(a|x_{tj}) e_c(a, x_{tj}) \ge d \end{aligned} \tag{21}$$

where x_{tj} denotes the j-th context sample randomly selected from historical observations and b denotes the total number of context samples. Since the arm selection policy for step t is needed, we make the convention that $x_{t1} = x_t$. Besides, e_r and e_c denote the UCBs of rewards and transactions, respectively, when using Lin-UCB and GLM-UCB. They denote the estimates of rewards and transactions, respectively, when using ensemble sampling.

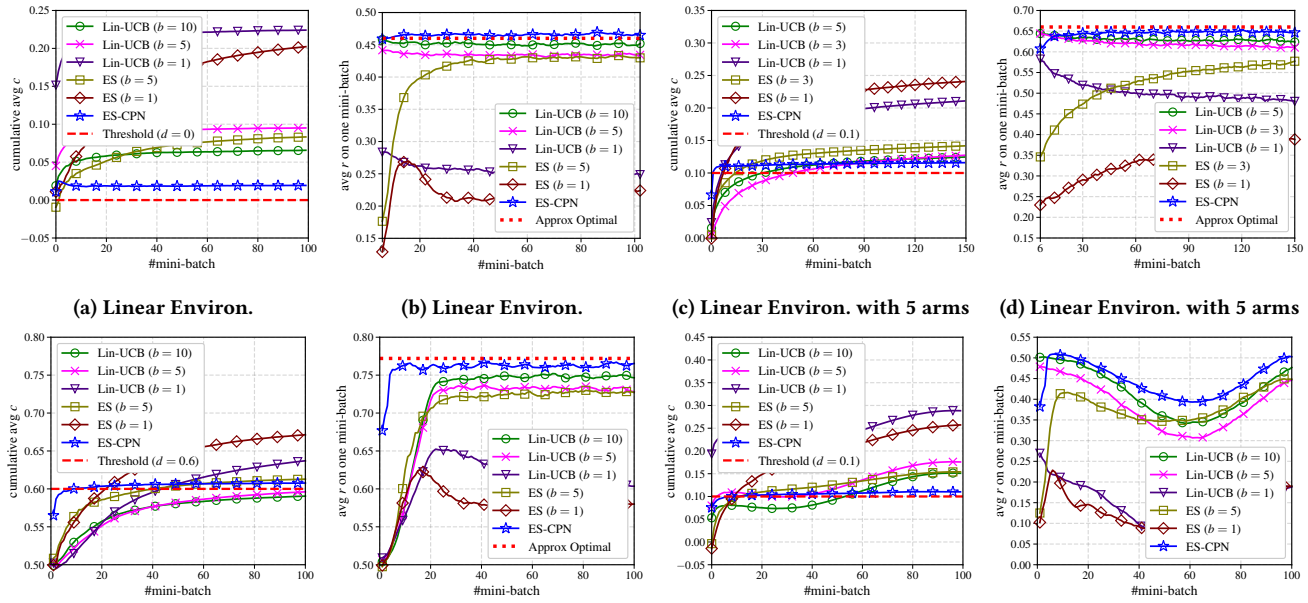
5.1 Synthetic Data

We test our algorithm by using the linear environment at first. More specifically, we let $r_t = x_t^{\mathrm{T}} \cdot \theta_r(a_t) + b_r(a_t) + \varepsilon_r$ and $c_t = x_t^{\mathrm{T}} \cdot \theta_c(a_t) + b_c(a_t) + \varepsilon_c$, where the context vector x_t is uniformly sampled from $[-1,+1]^{\dim(x)}$. ε_r and ε_c are Gaussian noise with mean 0.0 and variance 0.1. In Figures 2(a) and (b), we assume there are only two arms and $\dim(x) = 4$. We set the environment parameters as

$$\theta_r(1) = [-0.5, 0.5, -0.5, 0.5]\ ,\ \theta_r(0) = -\theta_r(1)\ ,\ b_r \equiv 0,$$
$$\theta_c(1) = [0.5, 0.5, -0.5, -0.5]\ ,\ \theta_c(0) = -\theta_a(1)\ ,\ b_c \equiv 0.$$

In Figures 2(c) and (d), we increase the number of arms to 5 and set the parameters of the extra three arms similarly to $\theta_r(1)$ and $\theta_c(1)$. In all these figures, for the convenience of display, we use one mini-batch to represent 1500 steps. For our algorithm, we set the number of policy networks $n = 10$ and $N = 150$, which means our policy networks are updated every 150 steps. All these policy networks are three-layer fully connected neural networks, where the hidden layer utilizes the ReLU activation function. For Figures 2(a) and (b), we set the number of neurons in the hidden layer as 8. For Figures 2(c) and (d) with more arms, we increase the neurons to 16. Meanwhile,

[1] In our problem, $T \to +\infty$, which means infinite horizon.



(a) Linear Environ. (b) Linear Environ. (c) Linear Environ. with 5 arms (d) Linear Environ. with 5 arms

(e) Bernoulli Environ. (f) Bernoulli Environ. (g) Non-Stationary Environ. (h) Non-Stationary Environ.

Figure 2: Experiments on synthetic data. Columns 1 and 3 show the cumulative average of c. Columns 2 and 4 show the average of r on one mini-batch (1500 steps). A good algorithm keeps the average of c higher than the threshold d and generate large r.

Table 1: Computation time of different algorithms when testing the linear environment in Figures 2(a) and (b)

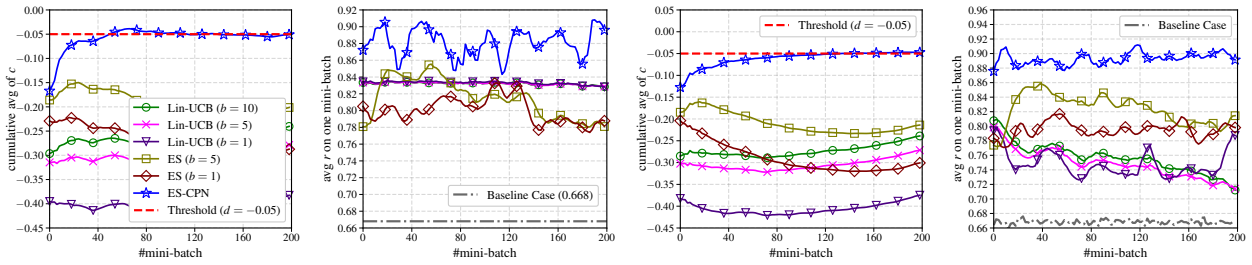
Algo.	ES-CPN	Lin-UCB			ES	
		$b = 1$	$b = 5$	$b = 10$	$b = 1$	$b = 5$
Time	**452s**	1085s	4374s	8872s	1716s	5378s

we let the risk aversion parameter $\rho = 2$, the step size of network updating $\delta = 0.1$, the number of training samples $N_{tr} = 1024$ and the memory size $N_{me} = 1500$. For ensemble sampling, we also set the number of neural networks as 10. The network uses two hidden layers. The settings of these hidden layers are the same as our policy network. We run all the algorithms for 10 times and show the mean of r and c in Figures 2(a)-(d).

From Figures 2(a) and (c), we can know that, in the simple linear environments, all algorithms successfully satisfy the constraint that the expectation of c must not be smaller than d. Then, from Figures 2(b) and (d), we can find that our algorithm, ES-CPN, significantly outperforms ensemble sampling. For Lin-UCB, in the linear environment, it can learn the correct values of parameters very quickly, which offers it great advantages. In fact, Lin-UCB represents the best performance that can be achieved by value-based bandit algorithms under the linear environment. However, even in this case, Lin-UCB cannot converge to the optimal policy because it can only use few samples of x to approximate the oracle via Equation 21. Otherwise, the computation time will become prohibitive, which we will discuss later. By contrast, our algorithm directly approaches the optimal policy based on online observations. This advantage makes our algorithm able to achieve higher rewards than Lin-UCB after a certain number of observations. Note that the t-test also supports the advantage of our algorithm on rewards. For example, for the end points in Figure 2(b), the p-value of the t-test between ES-CPN and Lin-UCB with $b = 10$ is only 0.00065, which means the differences are very significant.

To further investigate the gap between our algorithm and the oracle, we replace the estimates in Equation 21 with unnoised real values[2], randomly sample 200 samples of x, and solve the obtained linear programming problem via the simplex method. We run the above process for 1000 times and show the mean in Figures 2(b) and (d) as the dashed lines (Approx Oracle). In these figures, the reward curves of our algorithm are extremely close and even slightly higher than the dashed lines. This observation verifies that our algorithm successfully approaches the oracle. Another interesting observation to support our conclusion is that the c curve of our algorithm is higher than the threshold d with a tiny but very stable gap. This observation reveals that our algorithm always tries to decrease c as much as possible to get higher r. Moreover, we list the computation time of different algorithms on Xeon E5-2650 v2 in Table 1, which shows that our algorithm has significant advantages on computation efficiency. For Lin-UCB and ES, increasing the number of samples b can help to increase the rewards. Nevertheless, increasing b will also significantly increase the computation time. Thus, only small b is applicable in practice. Note that, when measuring the computation time of our algorithm, the updating and forwarding computation of all 10 policy networks are conducted sequentially. In practice, we can let the computation on different policy networks be parallel, which will further decrease the computation time.

[2] $e_r = x_t^{\mathrm{T}} \cdot \theta_r(a_t) + b_r(a_t)$ and $e_c = x_t^{\mathrm{T}} \cdot \theta_c(a_t) + b_c(a_t)$.



(a) (b) (c) (d)

Figure 3: Experiments on one of the largest e-commerce platforms in the world. (a) and (b) show the experiments on the offline data. (c) and (d) show the online experiments. The baseline cases in (b) and (d) mean when our regulating valve is not used.

In Figures 2(e) and (f), we use the Bernoulli environment to test our algorithm, where r and c both follow the Bernoulli distribution[3]. Bernoulli distribution is discrete, which is complementary to the continuous Gaussian noise in Figures 2(a)-(d). The settings of the environment parameters are the same as those in Figures 2(a). From Figures 2(e) and (f), we can get the same conclusions as those in Figures 2(a)-(d), which further verifies the advantages of our algorithm. Besides, we build a non-stationary environment based on Figures 2(a) and (b) by letting $b_r(0) = 0.1 \cdot \sin(\lambda t)$, $b_r(1) = 0.1 \cdot \cos(\lambda t)$ and $b_c(0) = b_c(1) = -0.1 \cdot \sin(\lambda t)$. Here, we set $\lambda = \pi/75000$ to make the whole figure with 100 mini-batches correspond to one period. The results shown in Figures 2(g) and (h) reveal that the advantages of our algorithm in the non-stationary environment are even larger than those in the stationary environment. This observation and the consistently good performance in Figures 2(a)-(f) show the good robustness of our algorithm to adapt to different environments, which is favorable in practical usage.

5.2 Real-World E-Commerce Platform

Next, we show the experiments on one of the largest e-commerce platforms in the world. We first build a static offline dataset by collecting 300, 000 query operations. The dataset contains the content of the web page shown to buyers (used to compute r), buyers' purchasing operations on the web page (used to compute c), and 74 features about buyers and their queries, such as, buyers' age and purchasing power index, the click-through rate and conversion rate of the query, etc. We directly use these raw data for experiments and demonstrate the results in Figures 3a and 3b. It is noted that adding the regulating valve and applying our algorithm can increase r from 0.668 in the baseline case to around 0.88 and meanwhile keep c around the desired lower bound $d = -0.05$. For the regulating valve, r means the percentage of buyer impressions received by honest sellers, and c denotes the rate of transaction changes. The buyer impression increment of honest sellers means the decline of fraud sellers. In other words, our regulating valve reduces the percentage of the buyer impressions received by fraud sellers from 0.332 to 0.12 (around 2/3 decline) and only losses 5% in transactions. Besides, it is also worth mentioning that our algorithm not only is the only algorithm satisfying the constraint on c but also can obtain the maximal r. The inferior performance of Lin-UCB and ES is caused by the skewed distribution of c. Buyers only purchase items on a very small ratio of web pages, which makes c a small value in most cases. Once buyers purchase something, there will be a large c signal. For Lin-UCB and ES, they are unable to capture the skewed distribution of c with only few samples when solving Equation 21. In this case, satisfying the constraint is infeasible, even though we have added the threshold d by the gap between the cumulative average of c and d as a complement.

Then, with the same settings, we directly conduct online experiments and show the results in Figures 3c and 3d. To keep the safety of the e-commerce platform, we actually divide buyers and queries into more than 30 channels. We choose one channel to test all algorithms and another channel to compute the baseline transaction per page, v_0. Since accumulating 30, 000 samples is very fast in the e-commerce platform, we can neglect the effects of time. The results of our online experiments are similar to the offline experiments. The only difference is that, due to the stochastic change of v_0, the convergence of the constraint becomes slower.

6 CONCLUSION

In this paper, we conduct the first study in e-commerce to regulate the buyer impressions of fraud sellers with constrained contextual bandits. To incorporate the constraint, we propose a novel policy-based bandit algorithm. It uses neural networks to represent arm selection policies, updates neural networks by solving a constrained local optimization problem, and avoids local optimum via ensemble sampling. We perform experiments on four synthetic environments and the data collected from one of the largest e-commerce platforms in the world. The results show that our algorithm achieves higher rewards with much less computation time than existing bandit algorithms. Besides, the experiments on real e-commerce data also show our algorithm reduces the buyer impressions of fraud sellers by 2/3, yet with only 5% transaction decline. Despite the extensive experiments, a missing part of our paper is to prove the regret upper bound of our algorithm, which is usually an essential part of bandit studies. However, for policy-based bandit algorithms with neural networks, there is no literature that we can refer to. It is very challenging to quantify the convergence rate of the update in Equations 16 and 18 as well as the effects of the ensemble sampling layer. Thus, we leave the regret study as our future work.

[3] $\Pr(r_t = 1) = \sigma[x_t^{\mathrm{T}} \cdot \theta_r(a_t) + b_r(a_t)]$, $\Pr(c_t = 1) = \sigma[x_t^{\mathrm{T}} \cdot \theta_c(a_t) + b_c(a_t)]$, where σ denotes the sigmoid function.

REFERENCES

[1] Yasin Abbasi-Yadkori, Dávid Pál, and Csaba Szepesvári. 2011. Improved algorithms for linear stochastic bandits. In *Proc. of NIPS*.

[2] Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. 2017. Constrained Policy Optimization. In *Proc. of ICML*.

[3] Shipra Agrawal, Nikhil R Devanur, and Lihong Li. 2016. An efficient algorithm for contextual bandits with knapsacks, and an extension to concave objectives. In *Proc. of COLT*.

[4] Shipra Agrawal and Navin Goyal. 2013. Thompson sampling for contextual bandits with linear payoffs. In *Proc. of ICML*.

[5] Stephen Boyd and Lieven Vandenberghe. 2004. *Convex optimization*. Cambridge university press.

[6] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. 2005. Learning to rank using gradient descent. In *Proc. of ICML*.

[7] Qingpeng Cai, Aris Filos-Ratsikas, Pingzhong Tang, and Yiwei Zhang. 2018. Reinforcement Mechanism Design for e-commerce. In *Proc. of WWW*.

[8] Qingpeng Cai, Aris Filos-Ratsikas, Pingzhong Tang, and Yiwei Zhang. 2018. Reinforcement Mechanism Design for Fraudulent Behaviour in e-Commerce. In *Proc. of AAAI*.

[9] Qingpeng Cai, Pingzhong Tang, and Yulong Zeng. 2018. Ranking Mechanism Design for Price-setting Agents in E-commerce. In *Proc. of AAMAS*.

[10] Wei Chu, Lihong Li, Lev Reyzin, and Robert Schapire. 2011. Contextual bandits with linear payoff functions. In *Proc. of AIStat*.

[11] Sarah Filippi, Olivier Cappe, Aurélien Garivier, and Csaba Szepesvári. 2010. Parametric bandits: The generalized linear case. In *Proc. of NIPS*.

[12] Yujing Hu, Qing Da, Anxiang Zeng, Yang Yu, and Yinghui Xu. 2018. Reinforcement Learning to Rank in E-Commerce Search Engine: Formalization, Analysis, and Application. In *Proc. of SIGKDD*.

[13] Ron Kohavi and Roger Longbotham. 2017. Online controlled experiments and a/b testing. In *Encyclopedia of machine learning and data mining*. Springer, 922–929.

[14] Andreas Krause and Cheng S Ong. 2011. Contextual gaussian process bandit optimization. In *Proc. of NIPS*.

[15] Lihong Li, Wei Chu, John Langford, and Robert E Schapire. 2010. A contextual-bandit approach to personalized news article recommendation. In *Proc. of WWW*.

[16] Ping Li, Qiang Wu, and Christopher J Burges. 2008. Mcrank: Learning to rank using multiple classification and gradient boosting. In *Proc. of NIPS*.

[17] Shichen Liu, Fei Xiao, Wenwu Ou, and Luo Si. 2017. Cascade Ranking for Operational E-commerce Search. In *Proc. of SIGKDD*.

[18] Xiuyuan Lu and Benjamin Van Roy. 2017. Ensemble Sampling. In *Proc. of NIPS*.

[19] Renxin Mao, Zhao Li, and Jinhua Fu. 2015. Fraud Transaction Recognition: A Money Flow Network Approach. In *Proc. of CIKM*.

[20] Yabo Ni, Dan Ou, Shichen Liu, Xiang Li, Wenwu Ou, Anxiang Zeng, and Luo Si. 2018. Perceive Your Users in Depth: Learning Universal User Representations from Multiple E-commerce Tasks. In *Proc. of SIGKDD*.

[21] Feiyang Pan, Qingpeng Cai, Pingzhong Tang, Fuzhen Zhuang, and Qing He. 2018. Policy Gradients for Contextual Bandits. *CoRR* abs/1802.04162 (2018). arXiv:1802.04162 http://arxiv.org/abs/1802.04162

[22] Reuven Y Rubinstein and Dirk P Kroese. 2016. *Simulation and the Monte Carlo method*. Vol. 10. John Wiley & Sons.

[23] Niranjan Srinivas, Andreas Krause, Sham M Kakade, and Matthias W Seeger. 2012. Information-theoretic regret bounds for gaussian process optimization in the bandit setting. *IEEE Transactions on Information Theory* 58, 5 (2012), 3250–3265.

[24] Ning Su, Yiqun Liu, Zhao Li, Yuli Liu, Min Zhang, and Shaoping Ma. 2018. Detecting Crowdturfing "Add to Favorites" Activities in Online Shopping. In *Proc. of WWW*.

[25] Wen Sun, Debadeepta Dey, and Ashish Kapoor. 2017. Safety-Aware Algorithms for Adversarial Contextual Bandit. In *Proc. of ICML*.

[26] Pingzhong Tang. 2017. Reinforcement mechanism design. In *Proc. of IJCAI*.

[27] Huasen Wu, R Srikant, Xin Liu, and Chong Jiang. 2015. Algorithms with logarithmic or sublinear regret for constrained contextual bandits. In *Proc. of NIPS*.

[28] Haitao Xu, Zhao Li, Chen Chu, Yuanmi Chen, Yifan Yang, Haifeng Lu, Haining Wang, and Angelos Stavrou. 2018. Detecting and Characterizing Web Bot Traffic in a Large E-commerce Marketplace. In *European Symposium on Research in Computer Security*.

[29] Chunqiu Zeng, Qing Wang, Shekoofeh Mokhtari, and Tao Li. 2016. Online context-aware recommendation with time varying multi-armed bandit. In *Proc. of SIGKDD*.

[30] Mengchen Zhao, Zhao Li, Bo An, Haifeng Lu, Yifan Yang, and Chen Chu. 2018. Impression allocation for combating fraud in e-commerce via deep reinforcement learning with action norm penalty. In *Proc. of IJCAI*.

[31] Martin Zinkevich, Markus Weimer, Lihong Li, and Alex J. Smola. 2010. Parallelized Stochastic Gradient Descent. In *Proc. of NIPS*.

[32] Masrour Zoghi, Tomas Tunys, Mohammad Ghavamzadeh, Branislav Kveton, Csaba Szepesvari, and Zheng Wen. 2017. Online Learning to Rank in Stochastic Click Models. In *Proc. of ICML*.