We Know What You Want: An Advertising Strategy Recommender System for Online Advertising

Liyi Guo[1], Junqi Jin[2], Haoqi Zhang[1], Zhenzhe Zheng*[1,3], Zhiye Yang[2], Zhizhuang Xing[2], Fei Pan[2]
Lvyin Niu[2], Fan Wu[1], Haiyang Xu[2], Chuan Yu[2], Yuning Jiang[2], Xiaoqiang Zhu[2]
[1]Shanghai Jiao Tong University, [2]Alibaba Group
[3]State Key Lab. for Novel Software Technology, Nanjing University
{liyiguo1995, zhanghaoqi39, zhengzhenzhe}@sjtu.edu.cn, fwu@cs.sjtu.edu.cn
{junqi.jjq, zhiye.yzy, zhizhuang.xzz, pf88537, lvyin.nly, shenzhou.xhy, yuchuan.yc, mengzhu.jyn,
xiaoqiang.zxq}@alibaba-inc.com

ABSTRACT

Advertising expenditures have become the major source of revenue for e-commerce platforms. Providing good advertising experiences for advertisers by reducing their costs of trial and error in discovering the optimal advertising strategies is crucial for the long-term prosperity of online advertising. To achieve this goal, the advertising platform needs to identify the advertiser's optimization objectives, and then recommend the corresponding strategies to fulfill the objectives. In this work, we first deploy a prototype of strategy recommender system on Taobao display advertising platform, which indeed increases the advertisers' performance and the platform's revenue, indicating the effectiveness of strategy recommendation for online advertising. We further augment this prototype system by explicitly learning the advertisers' preferences over various advertising performance indicators and then optimization objectives through their adoptions of different recommending advertising strategies. We use contextual bandit algorithms to efficiently learn the advertisers' preferences and maximize the recommendation adoption, simultaneously. Simulation experiments based on Taobao online bidding data show that the designed algorithms can effectively optimize the strategy adoption rate of advertisers.

CCS CONCEPTS

• **Information systems → Display advertising**; • **Applied computing → Electronic commerce**.

KEYWORDS

E-commerce; Display Advertisement; Advertising Strategy Recommendation

ACM Reference Format:
Liyi Guo, Junqi Jin, Haoqi Zhang, Zhenzhe Zheng, Zhiye Yang, Zhizhuang Xing, Fei Pan, Lvyin Niu, Fan Wu, Haiyang Xu, Chuan Yu, Yuning Jiang, Xiaoqiang Zhu. 2021. We Know What You Want: An Advertising Strategy Recommender System for Online Advertising. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '21), August 14–18, 2021, Virtual Event, Singapore.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3447548.3467175

1 INTRODUCTION

With the rapid development of the Internet, online e-commerce platforms, such as Amazon [19], eBay [29], and Taobao [42], have become the major venues for people to buy and sell products. E-commerce advertising is a critical marketing tool for advertisers to efficiently deliver their products to potential buyers, and the revenue from advertising has become the major source of income for e-commerce platforms [6, 7, 9, 17]. Thus, providing good advertising experiences for advertisers is crucial to the long-term prosperity of online advertising. It is highly necessary for platforms to fully understand the optimization objectives of advertisers, and launch advertising services to fulfill them.

As a representative e-commerce advertising platform, Taobao has established an advanced online advertising system with various intelligent services to decide the display of tens of millions of online ads. For the advertisers' side, in addition to traditional manual bidding, the Taobao display advertising system also provides auto-bidding services, such as Budget Constraint Bidding (BCB) and Multiple Constraint Bidding (MCB), to help advertisers fulfill their advertising objectives in a cost-efficient way. Since the performance achieved by the intelligent advertising strategies highly depends on the preset optimization objectives, the platform would like to first identify the advertisers' preference for various advertising performance. The current launching process of an ad campaign shown in Figure 1 contains three steps: (1) Choosing an optimization objective, such as the number of clicks, gross merchandise volume (GMV), etc. (2) Selecting the targeted users. (3) Choosing manual bidding or auto-bidding services. However, such a self-service launching process would introduce some obstacles due to the information asymmetry between advertisers and the

This work was supported in part by Science and Technology Innovation 2030 – "New Generation Artificial Intelligence" Major Project No. 2018AAA0100905, in part by China NSF grant No. 62025204, 62072303, 61902248, and 61972254, and in part by Alibaba Group through Alibaba Innovation Research Program, and in part by Shanghai Science and Technology fund 20PJ1407900. The opinions, findings, conclusions, and recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the funding agencies or the government.
*Z. Zheng is the corresponding author.

KDD '21, August 14–18, 2021, Virtual Event, Singapore

ACM ISBN 978-1-4503-8332-5/21/08…$15.00
https://doi.org/10.1145/3447548.3467175

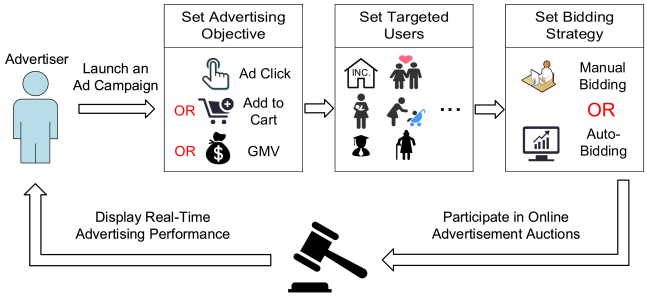


Figure 1: The advertising process in Taobao display advertising platform.

platform. On the one hand, the advertisers have no access to the real-time information of online ad auctions, which makes them difficult to set appropriate advertising strategies, leading to poor performance and the loss of budget. On the other hand, the current number of advertising objectives for advertisers to choose is limited, which does not capture all possible optimization objectives of advertisers. The current advertising system does not have channels for advertisers to reveal fine-grained optimization objectives. A recent survey reveals that a large number of new advertisers have left the platform because their advertising objectives were not well satisfied [11]. To solve the above problems and further improve the performance of the advertising platform, we initialize a new research direction: designing advertising strategy recommender systems for advertisers.

An intuitive approach to implement an advertiser-side recommender system is to borrow ideas from the extensive works in user-side recommender system, which focuses on the matching between products and users [4, 5, 10, 41, 43]. However, simply adopting these approaches to the advertiser-side recommendation would encounter several new challenges: (1) User-side recommendation applies standard supervised learning methods based on users' click or purchase behaviors, which are the direct and explicit signals for users to express their interests and preferences. While in the current advertising platforms, due to the lack of efficient and accurate estimation for advertising performance, the platforms do not provide the predicted performance of recommended strategies, and thus advertisers cannot express their preferences for advertising performance. (2) Most models on the user-side recommendation are based on a discrete corpus, that is the product library. However, on the advertiser-side, the corpus is often seen as a complex high-dimensional continuous space, which represents the potential performance results from different advertising strategies. (3) There are no available public data set yet for training the model of recommender systems for advertisers. It is also expensive for the recommender system to explore an advertiser's advertising objective, because the advertiser's adoption of new strategies would expend her advertising cost immediately.

To overcome the above challenges, we first design a prototype recommender system to help advertisers optimize their bid prices and select the targeted users. Online evaluations demonstrate that the prototype system can indeed increase the advertiser's advertising performance and the platform's revenue. We further augment this prototype recommendation system by the following three extensions. To solve the difficulty of learning advertisers' optimization objectives, we recommend the advertising strategies associated with the predicted performance, which is obtained through the ad auction simulator. We model the advertiser's preference as the weight for each individual performance indicator, and the resulting optimization objective is the linear combination of performance indicators. We learn the weights through the advertisers' interactions with the recommended strategies. For the second challenge, we design a neural network with the input of advertisers' features and their historical adapted strategies, to predict the advertisers' potential preferences, reducing the complexity of searching over continuous space. For the last challenge, to effectively learn advertisers' preferences and objectives at the same time, we consider the recommended strategies as bandits and the advertiser's strategy adoption as the environmental feedback, and leverage the techniques from contextual bandit to solve this problem.

We summarize the contributions of this work as follows:

- To the best of our knowledge, we are the first to investigate the problem of advertising strategy recommendation for online advertising. We define the objective of an advertiser is to maximize a linear combination of multiple performance indicators, where the weight for each performance indicator is considered as the advertiser's preference. We recommend the advertising strategies associated with the predicted performance, and use advertisers' adoption behaviors as signals to infer their preferences.
- To efficiently recommend advertising strategies, we should exploit the existing information to optimize the advertiser's adoption rate and properly explore the advertiser's underlying preference at the same time. We model this explore-and-exploit process as a contextual bandit problem: we regard the recommending advertising strategies as the actions and the advertisers' adoption behaviors as the rewards. We also use the Dropout technique to make a efficient trade-off between exploration and exploitation.
- We deploy a prototype system recommending bid prices and targeted users to advertisers on the Taobao display advertising platform. The online A/B test demonstrates the effectiveness of recommending advertising strategies for advertisers, increasing the advertiser's performance and the platform's revenue. We build an offline simulation environment based on Taobao online advertising bidding data, and conduct a series of experiments. The experiment results show that the strategy recommender system we design can effectively learn the advertisers' preferences and improve the recommendation adoption rate of advertisers.

The rest of the paper is organized as follows: In Section 2, we introduce related works about user-side recommender systems and real-time bidding algorithms. In Section 3, we introduce the prototype recommender system deployed in Taobao's advertising platform and the new strategy recommender system designed based on the prototype system. In Section 4, the learning algorithm of the new strategy recommender system is illustrated. The experimental results are given in Section 5. We conclude the work in Section 6.

2 RELATED WORK

2.1 Recommender System

In recent years, the recommendation problem has been extensively studied in both academia and industry, and different types of recommender systems have been deployed in various scenarios [4, 5, 10, 41, 43]. The recommender systems resolve the problem of information overload: help users discover useful information quickly and provide personalized services [22]. Several approaches, such as collaborative filtering [30, 31], content-based filtering [20, 27] and hybrid filtering [1], have been widely used in designing recommender systems. Collaborative filtering leverages the correlation between products to do recommendation. Content-based filtering recommends products to users based on their similarity, which is extracted from the user's historical choices, user profiles and product information. Hybrid filtering combines collaborative filtering and content-based filtering to improve the accuracy of recommendations.

Although extensive studies have been done on the user-product recommendation, there are two main differences between the user-side recommendation and the advertiser-side recommendation. First, user-product recommendations are usually based on a unified product library for feature extraction and correlation analysis. However, in the advertiser-side recommendation, the recommended product is an abstract advertising strategy (such as bidding strategy), which is hard to analyze simply through feature extraction. Second, advertiser-side recommendations should not only solve the problem of information overload but also help advertisers optimize their advertising performance, achieving the promised performance of the recommended strategies. Therefore, we need to jointly consider the advertising strategy recommendation and advertising objectives optimization, which is related to real-time bidding.

2.2 Real-Time Bidding

In the real-time bidding scenario, researchers have conducted an in-depth study on the problem of auto-bidding to optimize certain optimization objectives [24, 28, 40]. Designing a real-time bidding algorithm is formulated as an online optimization problem with an optimization goal, e.g. maximize GMV under the budget constraint [35, 39]. Zhang et al. proposed an offline optimal bidding strategy by bidding $\frac{v}{\lambda}$ for each impression under the second-price auction mechanism, where v is the value of the impression, and λ is a fixed parameter determined by the environment [37]. Wu et al. further applied a model-free reinforcement learning method to adjust the λ in an online manner, aiming to pace the spend of the budget within the real-time environment [35]. The subsequent research efforts attempted to extend the single objective and single constraint to the setting with various optimization goals under multiple constraints [15, 36, 38]. Zhang et al. used feedback control methods to adjust bidding based on real-time performance, and satisfied auction winning rate and pay per click (PPC) constraints [38]. Yang et al. formulated the GMV optimization problem with budget and PPC constraints as an online linear programming problem [36].

Based on the extensive work about real-time bidding, the auto-bidding tools provided by the platform can well handle the optimization problem as long as the objectives are known in advance. However, in a display advertising system, advertisers often have a variety of specific objectives for different ads at different marketing phases, leading to different optimization goals. For example, advertisers would like to maximize the amount of impressions for new products, and maximize the GMV of the product in promotional sessions. However, there still does not exist effective ways for the advertising platforms to figure out the advertiser's advertising objectives. This significantly degrades the performance of the real-time bidding algorithms.

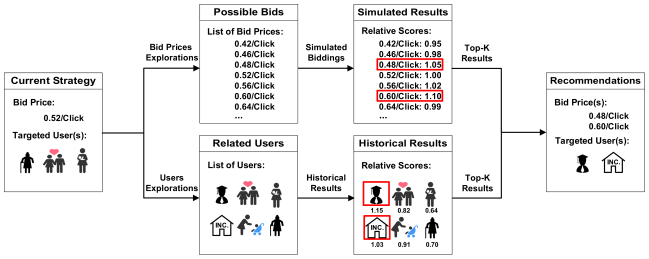


Figure 2: The prototype recommender system: a bid optimization module and a targeted user optimization module.

3 SYSTEM DESIGN

In this section, we first illustrate our currently deployed advertising strategies recommender system for advertisers, and formally define the advertisers' preference and optimization goals. We then introduce our extensions for this prototype system. Finally, we formulate the problem of advertising strategy recommendation as the contextual bandit, and provide efficient solutions to solve it.

3.1 Prototype Recommender System

To reduce the expensive trial and error costs for advertisers to look for optimal advertising strategies, and explore potential marketing opportunities, we have already deployed a recommender system on the Taobao display advertising platform as shown in Figure 2. The system consists of two components: a *bid optimization module* and a *targeted user optimization module*. The bid optimization module recommends bids for advertiser based on her selected targeted users and the pre-defined optimization goal, and the targeted user optimization module explores potential users for the advertiser. The prediction ability of the system is supported by the *ad auction simulator*, which estimates the advertising performance of possible strategies using the advertiser's historical bidding logs through conducting simulated auctions.

Bid Optimization Module. To search for high-quality bids for a certain type of targeted users, the bid optimization module first samples some candidate bids, predicts the advertising performance of these bids through the ad auction simulator. The module ranks these candidate bids by some rule, such as the relative increment on CTR (or CVR) compared with the average CTR (or CVR) of the historical adopted bidding strategies. The module lists the top-K bidding strategies associated with some other information, such as historical advertising performance, current bids and predicted advertising performance.

Targeted User Optimization Module. This module explores potential users for advertisers. It first merges similar types of users

in a proper way, such as the users selected by advertisers belong to the same product category. The module calculates the average historical KPIs of these merged users, ranks them based on some performance indicators (e.g. *CTR*, *PPC*, *CVR*, *GMV*). For each ad campaign, the top-K targeted users would be recommended, and the predicted number of unique visitors for the ad campaign covered by these targeted users would also be shown.

The prototype recommender system increases the ad platform's Average Revenue per User (ARPU) by 1.2% in the online A/B test in May 2020. The implementations of the system, experimental settings, and results of the online evaluations are shown in Section 5.1. The currently deployed system is still in the infant stage: (1) The system ranks the candidate advertising strategies based on some deterministic metrics (i.e. relative increment in some KPIs or a fixed weighted average over KPIs). However, advertisers have various advertising preferences and objectives over the advertising performance, and ranking recommended strategies based on this information would improve the adoption rate and provide personalized advertising experiences for advertisers. (2) The current system gives recommendations based on a fixed bid price. We observe that the advertisers care more about the performance of the advertising strategy, instead of the specific contents of the strategy, such as the bid for each auction. We argue that the strategy recommender system for advertisers should focus on the advertising performance.

3.2 Advertising Strategy Recommender System

To further augment the current prototype system, we first formulate the problem of strategy recommendation, and then we design a new recommender system based on the formulation. We define the performance of an ad campaign as: $\mathbf{v} = [v_1, v_2, \cdots, v_n]^\top$, where n represents the number of KPIs, and v_i represents the value of the i-th KPI. Since advertisers have different preferences over different KPIs, we define the advertiser's preference as a weight vector: $\mathbf{w} = [w_1, w_2, \cdots, w_n]^\top$, where w_i is the weight for the i-th KPI. For a specific ad campaign, suppose that the advertiser's preference is $\mathbf{w}^*$, and the corresponding optimization objective of the advertiser is $\mathbf{w}^{*\top} \cdot \mathbf{v}$, which is a weighted sum of advertising performance. By defining Π as the advertiser's bidding strategy, for the advertiser with the preference vector $\mathbf{w}^*$, recommending the optimal bidding strategy $\Pi_{\mathbf{w}^*}$ is to solve the following optimization problem:

$$\Pi_{\mathbf{w}^*} = \arg\max_{\Pi} \mathbf{w}^{*\top} \cdot \mathbf{v}_{\Pi}, \tag{1}$$

where $\mathbf{v}_\Pi$ is the advertising performance that the bidding strategy Π can achieve. To solve optimization problem in (1), an intuitive solution is to use some real-time bidding algorithms [3, 35, 38, 39] with the preference vector $\mathbf{w}^*$. Therefore, the preference vector $\mathbf{w}^*$ guides the advertising platform to obtain the most satisfactory advertising strategies and performance for advertisers. However, due to the information asymmetry between the platform and advertisers, we cannot directly know the vector $\mathbf{w}^*$.

To overcome the above obstacles, we design a new recommender system framework shown in Figure 3 to learn about the advertisers' preference and then the optimization objective based on the interaction between advertisers and the recommender system. For each request of advertising strategy recommendation from advertisers,

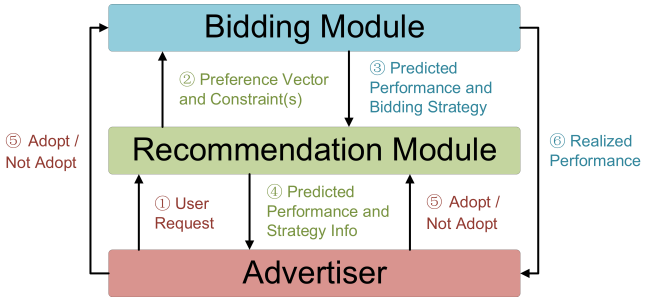


Figure 3: The architecture of an advertising strategy recommender system.

the recommender system processes as follows: (1) The *recommendation module* receives the request from an advertiser and obtains her ad campaigns' information. (2) The *recommendation module* generates an estimated preference vector $\mathbf{w}$, which is sent, together with the constraint information such as the ad campaign's remaining budget, to the *bidding module*. The *recommendation module* inquires about the potential advertising performance for this preference vector. (3) The *bidding module* generates the optimal bidding strategy $\Pi_{\mathbf{w}}$ based on the estimated preference vector and constraint(s), and returns the predicted advertising performance under this bidding strategy, $\mathbf{v}_{\Pi_{\mathbf{w}}} = [v_1, v_2, \cdots, v_n]^\top$ to the *recommendation module*. (4) The *recommendation module* displays the bidding strategy $\Pi_{\mathbf{w}}$ and the corresponding advertising performance $\mathbf{v}_{\Pi_{\mathbf{w}}}$ to the advertiser, which is easier to understand than only recommending the bidding strategy. (5) The advertiser chooses to adopt the recommending strategy or not by checking the advertising performance, and the *recommendation module* collects the feedback from the advertiser. At the same time, once the advertiser adopts a recommendation, the *bidding module* would conduct the real-time bidding algorithm based on the recommended strategy, to fulfill the promised advertising performance. (6) The *bidding module* sends the realized advertising performance to the advertisers.

According to the above recommendation process, when the platform makes a bidding strategy recommendation to the advertiser, the advertiser would adopt the strategy if she is satisfied with the expected performance of the strategy; otherwise, the advertiser would reject. The adoption behaviors reflect advertisers' preferences and interests in the recommended advertising strategy and the associated performance. By efficiently learn the advertiser's preference, the platform can maximize the adoption rate. However, for a newly developed recommender system, a significant number of advertisers are likely to be entirely new without any historical adoption record, which is known as a *cold-start* problem [26]. Thus, it is necessary to explore the advertisers' preferences in a continuous space for new advertisers. In the advertising platform, acquiring such information need to evaluate the advertisers' responses to different advertising strategies, which could be expensive and might reduce the advertiser's satisfaction in the short term. Hence, this raises the problem of balancing two competing goals: maximizing the advertiser's satisfaction in the short term, and gathering information about advertiser's preference for the long-term development of the recommender system.

The above problem of the trade-off between two competing goals can be formulated as a feature-based exploitation/exploration process. Specifically, in this work, we leverage the technique from *contextual bandit*, where an agent recommends the advertising strategy sequentially based on the contextual information of the ad campaigns while adapting its action selection strategy simultaneously based on the advertisers' feedback. We illustrate our modeling in detail in the next subsection.

3.3 Contextual Bandit Modeling

In this subsection, we formulate the contextual bandit problem to model the advertising strategy recommendation. We formally define the notions of *agent*, *state*, *action* and *reward* in the context of strategy recommendation. The *agent* is the strategy recommender system, and all advertisers visiting the system constitute the environment. When an advertiser enters the recommender system, the agent needs to estimate an appropriate preference vector $\mathbf{w}$ for each of the advertiser's ad campaigns based on the advertiser's information and historical behaviors of strategy adoption observed by the agent. With the advertiser's preference vector, we can derive the optimal bidding strategy and the corresponding advertising performance through the ad auction simulator. Then, for each ad campaign, the agent displays the recommended bidding strategy and estimated advertising performance to the advertiser. The advertiser's adoption behavior (adopt or reject the recommendation) can be regarded as the *reward* of the environment to the agent. We summarize the state $\mathcal{S}$, action $\mathcal{A}$, and reward $\mathcal{R}$ of the contextual bandit problem as follows:

(1) Status $\mathcal{S}$: Information related to the ad campaign that the platform can observe, such as the ad campaign's information, the advertiser's historical adoption-related information and etc. We use a feature vector $\mathbf{x}$ to represent the state $\mathcal{S}$.

(2) Action $\mathcal{A}$: Estimating the advertiser's preference vector $\mathbf{w}$. The agent obtains the (estimated) optimal bidding strategy and the corresponding advertising performance, by sending the constraint information in the state $\mathbf{x}$ and the preference vector $\mathbf{w}$ to the bidding module.

(3) Reward $\mathcal{R}$: We set the reward to be 1 when the advertiser adopts the recommended advertising strategy; otherwise the reward is 0. The expected reward $\mathcal{R}(\mathcal{S}, \mathcal{A})$ represents the expected adoption rate under state $\mathcal{S}$ and action $\mathcal{A}$. That is the reward of the action is the advertiser's adoption feedback, and we need to maximize the reward by recommending the strategy most likely to be adopted by the advertiser.

Based on the above modeling, the agent continuously recommends strategies for visiting advertisers. We number the discrete recommendation rounds as $t = 1, 2, 3, \ldots$. In the t-th round, the contextual bandit algorithm works as follows:

(1) The agent observes the feature vector $\mathbf{x}_t$ of the current ad campaign.

(2) Based on $\mathbf{x}_t$, the agent predicts a preference vector $\mathbf{w}_t$, and sends the constraint information in $\mathbf{x}_t$ together with $\mathbf{w}_t$ to the bidding module, which performs the simulated bidding and returns the result $\mathbf{v}_t$. Then the agent recommends $\mathbf{v}_t$ to the advertiser and gets the reward $r_{t,\mathbf{w}_t}$.

(3) The observation of the current round $(\mathbf{x}_t, \mathbf{w}_t, \mathbf{v}_t, r_{t,\mathbf{w}_t})$ is used to help the agent make its future decision.

In the above process, the *T-trial Payoff* for T rounds recommendations is $\sum_{t=1}^{T} r_{t,\mathbf{w}_t}$. Similarly, we define the *Optimal Expected T-trial Payoff* as $\mathbf{E}\left[\sum_{t=1}^{T} r_{t,\mathbf{w}_t^*}\right]$, where the preference vector $\mathbf{w}_t^*$ is the true preference of the advertiser in the t-th round, and the optimal bidding strategy based on $\mathbf{w}_t^*$ would get the maximum advertising performance for advertisers. Our goal is to maximize the *Expected T-trial Payoff* or equivalently, we can also regard the goal as minimizing the *Expected T-trial Regret* $R(T)$ for T rounds recommendations as follows:

$$R(T) \quad \stackrel{\text{def}}{=} \quad \mathbf{E}\left[\sum_{t=1}^{T} r_{t,\mathbf{w}_t^*}\right] - \mathbf{E}\left[\sum_{t=1}^{T} r_{t,\mathbf{w}_t}\right]. \tag{2}$$

The *Expected T-trial Regret* $R(T)$ can be interpreted as the gap between the expected adoption amount of the optimal recommendation strategy and our designed recommendation strategy under T rounds of recommendations, and $\frac{R(T)}{T}$ is the gap between the expected adoption rate of the optimal recommendation strategy and the actual recommendation strategy.

4 ALGORITHM DESIGN

In the classic contextual bandit algorithm, the agent chooses to pull the arms based on some strategies to jointly learn the expected reward of each arm and the accumulated reward of the selected arms over a period. In the classic contextual bandit, the number of possible actions is usually discrete and finite, while the action space (the possible values of the preference vector) in advertising strategy recommendation is a high-dimensional continuous space. Furthermore, for each preference vector, it is also time consuming to solve a large scale of linear programming to obtain the bidding strategy and the advertising performance in the ad auction simulator. With regard to this issue, we cannot output each action's estimated reward at the same time, which makes it difficult to directly apply the existing bandit algorithms such as ϵ-greedy or upper confidence bound [16]. To solve the above problem, we divide the reward learning process into two steps: (1) Choosing the action based on the observed information. (2) Establishing the relation between (the estimated bidding strategy and the advertising performance) and the advertiser's adoption rate.

We first build the connection between the advertiser information and the advertising preference. More specifically, based on the action selection strategy, we obtain the preference vector $\mathbf{w}$ under the state $\mathcal{S}$ that the platform can observe as:

$$\begin{aligned} \mathbf{w} &= f(\mathbf{x}) \\ &= f(advertiser\ behaviors, ad\ profile, scenario, etc.), \end{aligned} \tag{3}$$

where the function f is a mapping between the environmental state S and the preference vector $\mathbf{w}$, the input $\mathbf{x}$ of f is the representation of the state $\mathcal{S}$, and the output of the network is $\mathbf{w}$. We use a multi-layer perceptron model (MLP) to build such a mapping. The label of the neural network (i.e. the reward of the action) is the adoption

behavior of the advertiser. We recall that $\mathbf{v}$ is the advertising performance under the preference vector $\mathbf{w}$, and the value of $\mathbf{w}^\top \cdot \mathbf{v}$ is the utility that the advertiser can obtain. Since the advertiser's adoption rate is positively correlated with $\mathbf{w}^\top \cdot \mathbf{v}$, we model the relation of $\mathbf{w}^\top \cdot \mathbf{v}$ and the adoption rate as follows:

$$p(\text{Adopt}) = \sigma(\mathbf{w}^\top \cdot \mathbf{v}), \tag{4}$$

where σ is the sigmoid function, and the advertising performance $\mathbf{v}$ based on $\mathbf{w}$ is also part of the model input. In practice, advertising performance $\mathbf{v}$ could be normalized by the advertiser's budget since we are only interested in the relative value in $\mathbf{v}$ and $\mathbf{w}$.

Based on the above discussion, the estimation of the action value (the preference vector $\mathbf{w}$) of the network can be updated through gradient descent. For each iteration, the model first observes the environmental feature $\mathbf{x}$ and estimates the preference $\mathbf{w}$, predicts the advertising performance $\mathbf{v}$ according to $\mathbf{w}$, and calculates the adoption rate $p(\mathbf{x}, \mathbf{v})$. Then, we update the parameters of the model through the following loss function:

$$L = -\frac{1}{N} \sum_{(\mathbf{x},\mathbf{v},y)\in\mathcal{D}} (y \times logp(\mathbf{x}, \mathbf{v}) + (1-y) \times log(1 - p(\mathbf{x}, \mathbf{v})), \tag{5}$$

where the set $\mathcal{D}$ is the data set of size N in this update iteration, and the label y is the advertiser's realized adoption behavior.

Action selection strategy. In the context of advertising strategy recommendation, exploration means recommending strategies based on new possible preference vectors to explore the potential interests of advertisers, and exploitation means recommending corresponding strategies based on the currently learned preference vector $\mathbf{w}$, which is the output of the model. Thompson Sampling is one of the efficient ways to make the trade-off between the exploitation and exploration [34]. Generally speaking, Thompson Sampling requires Bayesian processing of model parameters. In each step, Thompson Sampling samples a new set of model parameters, and then selects the preference vector based on the model with the sampling parameters. This can be seen as a kind of random hypothesis test: the more likely parameters are sampled more frequently and thus be rejected or adopted more quickly. Specifically, the process of Thompson Sampling is as follows:

(1) Sampling a new set of parameters for the model.
(2) Selecting the preference vector with the highest expected reward based on the model with the sampling parameters.
(3) Updating the model and going back to (1).

Performing Thompson Sampling in a neural network needs to describe the uncertainty of the network. Bayesian models [2, 23, 25] offer a mathematically grounded framework to represent the uncertainty of the models, but usually, come with a prohibitive computational cost [8]. Dropout, a simple yet effective way to avoid neural networks from overfitting [33], is proved to be a Bayesian approximation representing the uncertainty of the deep learning model [8]. Inspired by the above discussion, we apply Dropout in neural networks to balance exploration and exploitation in our recommendation problem.

Table 1: The advertising performance of the prototype recommender system in the online A/B test.

Metric	Result	Metric	Result
ARPU	+1.2%	RPM	+0.8%
Click Number	+1.5%	Payment Number	+3.3%
CTR	+1.0%	Payment Amount	+4.3%
Cost	+1.3%	CVR	+1.9%
ROI	+3.1%	PV	+0.5%

5 EXPERIMENTS

In this section, we first evaluate our prototype recommender system through online evaluations, and then build a simulation environment to conduct extensive simulated evaluations of the proposed advertising strategy recommendation methods. We summarize our experimental results as follows: (1) The online evaluations demonstrate the potential benefits of introducing advertising strategy recommendation toward advertisers. The recommender system not only optimizes the advertiser's advertising performance but also increases the revenue of the platform. (2) The designed neural network is effective in the advertiser adoption rate optimization, which depends on the accurate prediction of advertiser preferences. (3) The Dropout trick can efficiently weigh the pros and cons between exploiting the existing preference information and exploring the potential preferences of advertisers. (4) We verify the generalization ability of the bandit algorithm through ablation studies.

5.1 Online Evaluations

System Implementation. We have deployed our prototype recommender system mentioned in Section 3.1 in the Taobao display advertising platform since February 2020. The detailed implementation of the system includes three main components: First, we maintain two individual databases for the bid optimization module and the targeted user optimization module, which store recommended items (e.g. targeted users and corresponding bids) for different ads. The recommended items in the two databases are updated daily. Second, we implement a ad auction simulator, which uses advertisers' bidding logs sampled from online auctions to predict the advertising performance using a specific bidding strategy with certain optimization objectives and constraints. Finally, we maintain a database that stores the realized advertising performance of the advertisers.

When an advertiser issues a request of strategy recommendation, the recommender system selects the recommended strategies based on the bandit algorithm, and predicts the advertising performance of these strategies using the ad auction simulator. Then the system ranks the recommended strategies based on some rules on the advertising performance and shows the top-K recommendations to the advertiser, together with the corresponding advertising performance. Once the advertiser adopts some recommendations, the system would carry out the adopted recommended strategy immediately.

Evaluation Results. In our online experiment between 2020-05-14 and 2020-05-27, we carefully select advertisers with the same consumption level based on their historical Average Revenue per User

(ARPU), the daily cost of advertisers in our experiments constitutes about 30% of the total daily cost of all the advertisers in the platform. It is worth to note that none of the advertisers in our experiment had used the recommender system before 2020-05-14. In the A/B test, half of the advertisers using the recommender system in the evaluation period form the test group, and the other half forms the control group. The average open rate of the bid optimization module and targeted user optimization module in the test group is 15.8% and 25.4% respectively, indicating that some advertisers are willing to use our recommender system. The average adoption rate of the advertisers is 4.5% and 3.8% respectively, for bid recommendation and targeted user recommendation. We find that advertisers would select and adopt a few recommendations carefully and cautiously, revealing the opportunity to introduce a learning module for personalized recommendations. The ARPU in the test group increases by 1.2% compared with the control group, showing that advertisers are satisfied with the recommended strategies and are willing to invest more in advertising, and thus increase the revenue of the platform. Moreover, as shown in Table 1, the overall advertising performance of ad campaigns in the test group is better than those in the control group, which verifies that the recommender system can indeed improve advertisers' advertising performance.

It could be shown that the above experimental results verify the effectiveness of the prototype recommender system. However, some weaknesses still exist in the infant system: (1) Advertisers need to select the recommendations they want from a large number of recommendations. (2) The improvement of advertising performance for advertisers is not significant. Consequently, the prototype recommender system needs to be augmented through extensions.

5.2 Simulation Settings

Compared with machine learning in a standard supervised environment, evaluations of bandit algorithms are more difficult [18]. Besides, bandit algorithms usually require a lot of interaction with the environment, making it costly to conduct experiments in a real advertising environment due to the large expenses of trial and error [32]. Therefore, most of the previous researches verify the effectiveness of the algorithm by building a simulation environment [12, 13]. In this section, we introduce the design of our simulation environment, which are the *advertiser module* and *bidding module* of the system architecture in Figure 3. The advertiser module simulates the interaction between the advertisers and the recommendation module in the real advertising system. The bidding module obtains the expected advertising performance based on ad campaigns' bidding logs.

Bidding module: The bidding module optimizes the objective function $\mathbf{w}^\top \cdot \mathbf{v}$ under the budget constraint. The advertising indicators in the simulation environment are PV, Click Number, and GMV. We use offline linear programming [37] to derive the optimal bidding strategy and the corresponding advertising performance based on the ad campaign's bidding logs .

Advertiser module: In the simulation environment, each ad campaign has a budget B and a preference vector $\mathbf{w}^*$. We generate the advertiser's preference vector $\mathbf{w}^*$ as $\mathbf{w}^* = g(\text{Parameter Set})$, where g is a function based on a parameter set, which represents the advertiser characteristics observed by the platform. Specifically, we design the following scheme: given the feature matrix composed of the m typical (basic) preference vectors $A_{n\times m} = [\mathbf{w}_1; \mathbf{w}_2; ...; \mathbf{w}_m]$ and the advertiser's feature vector $\mathbf{c} = [c_1, c_2, ..., c_m]^\top$, where c_i means the weight of the i-th typical preference $\mathbf{w}_i$ in the feature matrix. The preference vector of the advertiser is obtained by $\mathbf{w}^* = A \cdot c$. In the simulation, the agent can observe the feature matrix A and the feature vector $\mathbf{c}$. It is worth to note that other relations can also be designed and tested, but we cannot know the relation in the production environment in advance.

We next model the advertiser's adoption behavior. We apply the conditional logit model that is widely used in user selection modeling [21] to simulate the adoption behaviors of advertisers. In the conditional logit model, the probability that user i selects item j in the recommended set Ω is $P(j \mid \Omega) = \frac{e^{u(x_{ij})}}{\sum_{\ell\in\Omega} e^{u(x_{i\ell})}}$, where $u\left(x_{ij}\right)$ is the utility of the commodity j to the user i. The utility of the user selecting a "null" commodity (not selecting any commodity) is usually expressed by a constant [13]. Therefore, we design the advertiser's probability to adopt the recommended strategy in the simulation environment as:

$$\Pr(\texttt{Adopt}) = \frac{e^{u(\mathbf{w}^*,\mathbf{v},\mathbf{v}')}}{e^{u(\mathbf{w}^*,\mathbf{v},\mathbf{v}')} + C}, \tag{6}$$

where the vector $\mathbf{w}^*$ represents the advertiser's preference, and the vector $\mathbf{v}$ represents the advertising performance obtained through bidding according to the advertiser's preference vector $\mathbf{w}^*$. The vector $\mathbf{v}'$ represents the advertising performance obtained through bidding under the model's estimated preference vector $\mathbf{w}$, and the value C represents the utility of advertiser's rejection. The $u(\mathbf{w}^*, \mathbf{v}, \mathbf{v}')$ represents the utility function of the advertiser's adoption, we design $u(\mathbf{w}^*, \mathbf{v}, \mathbf{v}') = \alpha \times \frac{\mathbf{w}^{*\top}\cdot\mathbf{v}}{\mathbf{w}^{*\top}\cdot\mathbf{v}-\mathbf{w}^{*\top}\cdot\mathbf{v}'}$, where α is the parameter for adjusting the overall adoption rate of advertisers in the simulation environment. In the utility function, the term $\frac{\mathbf{w}^{*\top}\cdot\mathbf{v}}{\mathbf{w}^{*\top}\cdot\mathbf{v}-\mathbf{w}^{*\top}\cdot\mathbf{v}'}$ indicates the reciprocal of $\frac{\mathbf{w}^{*\top}\cdot\mathbf{v}-\mathbf{w}^{*\top}\cdot\mathbf{v}'}{\mathbf{w}^{*\top}\cdot\mathbf{v}}$, which is the relative utility gap between the advertising performance under the advertiser's estimated preference $\mathbf{w}$ and true preference $\mathbf{w}^*$. To prevent division by zero, we set the denominator greater than or equal to a small positive number ϵ. It can be seen from (6) that when the utility of the recommended strategy is closer to the optimal utility, the advertiser's adoption rate is higher, which is convenient for us to compare the effectiveness of different models.

In the initialization phase of the advertiser module, several ad campaigns with corresponding preference vectors and budgets are generated following the above procedure, and the visit event of the recommender system is randomly triggered. The ad campaigns in the simulation environment share the historical bidding logs.

Evaluation metrics and training parameters: The optimization goal of the contextual bandit algorithm is the *Expected T-trial Regret* in T rounds of interactions shown in (2): the accumulated expected regret is $\mathbf{E}\left[\sum_{t=1}^{T} r_{t,\mathbf{w}_t^*}\right] - \mathbf{E}\left[\sum_{t=1}^{T} r_{t,\mathbf{w}_t}\right]$. The accumulated adoption rate is $\frac{\sum_{t=1}^{T} r_{t,\mathbf{w}_t}}{T}$. We can use these two metrics to evaluate the performance of the model.

In the simulation experiments, input features of the model are preference-related features and advertiser's adoption behaviors related to the ad campaign. For preference-related features, we concatenate them together as one of the inputs. For advertiser's

Table 2: The advertising performance of the same ad campaign under different preferences. The values of each column are normalized by the maximum value in each column.

Objective Type	PV	Click Number	GMV
Optimizing PV	**1.0000**	0.7747	0.6751
Optimizing Click Number	0.7825	**1.0000**	0.8269
Optimizing GMV	0.7377	0.8454	**1.0000**

Table 3: Comparisons of different models. The values of each column are normalized by the maximum value in each column. Metric 1 indicates accumulated expected regret, and Metric 2 indicates accumulated adoption rate.

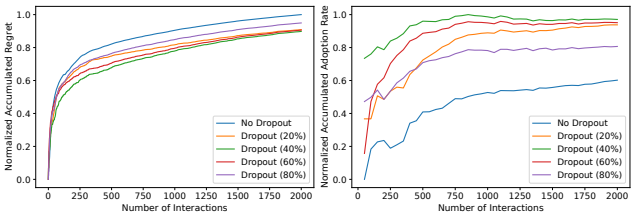
Model	Metric 1	Metric 2
Random Preference (No Learning Module)	1.0000	0.7606
No Dropout	0.7429	0.8239
Dropout (40%, No Preference-Related Info)	0.7195	0.8535
Dropout (80%)	0.5384	0.9144
Dropout (20%)	0.4154	0.9725
Dropout (60%)	0.4078	0.9817
Dropout (40%)	**0.3881**	**1.0000**

adoption behaviors, we apply average pooling to the corresponding model output $\mathbf{w}$ in historical adoption behaviors related to the ad. We use mini-batch SGD to train the model when it interacts with the environment, and use Adam [14] as the optimizer. To prevent the imbalance of the positive and negative samples to affect the model performance, we set the ratio of positive and negative samples in each training batch to 1 : 1.

5.3 Experimental Results

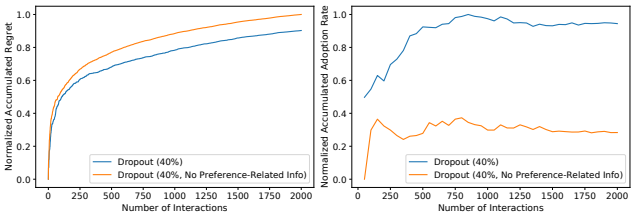
Exploration of advertiser's advertising performance space: First, we briefly explore the advertising performance space of the advertisers in the Taobao online advertising environment. In this experiment, we select three types of typical objectives of advertisers in the platform: maximizing total impressions, maximizing total clicks, and maximizing GMV. Suppose the advertising performance indicators are PV, Click Number and GMV, the preference vectors of these three types of advertisers are $[1, 0, 0]^\top$, $[0, 1, 0]^\top$ and $[0, 0, 1]^\top$, respectively. We conduct the optimal bidding strategy under the budget constraint for the above three preference vectors in the ad auction simulator. The experimental results are shown in Table 2, it can be seen that the performance of indicators to be optimized has been significantly improved under the corresponding preference vectors, which demonstrates the importance of understanding the advertisers' preferences in advertising performance optimization.

Comparison experimental results: We show the effectiveness of the proposed contextual bandit algorithm through comparison experiments, in which we compare models with different Dropout ratios or without Dropout. We also implement a recommendation strategy with a random preference. In each experiment, the agent interacts with the environment for 2000 rounds, and updates accumulated expected regret and accumulated adoption rate periodically.



(a) Accumulated expected regret. (b) Accumulated adoption rate.

Figure 4: Comparisons of models with different Dropout ratios.



(a) Accumulated expected regret. (b) Accumulated adoption rate.

Figure 5: Impact of preference-related information.

We show the experimental results in Table 3. From Table 3, we observe that the recommendation with a random preference causes huge degradation in the adoption rate, which motivates the necessity of considering advertiser preferences when recommending strategies. As shown in Table 3, even without using the Dropout trick, a model that explicitly learns the advertiser's preference can reduce the accumulated expected regret by 25.71% compared to the recommendation strategy without a learning module. From Table 3, we also learn that the context bandit algorithms which apply Dropout are more effective than the algorithm without Dropout. We observe that as the Dropout ratio increases (from 20%, 40%, 60% to 80%), the performance of the model first increases and then decreases. This is because (1) when the Dropout ratio is low, the model adopts a conservative exploration strategy. (2) when the dropout ratio is high, the model frequently explores the action space, which cannot make full use of the learned knowledge, resulting in performance degradation.

In Figure 4, we show the curves of accumulated expected regret and accumulated adoption rate for the models with different Dropout ratios in various numbers of interactions. To better evaluate the performance difference after the model converges, we normalize the accumulated regret by using the function $y = log(x + 1)$. From Figure 4, we observe that different models converge to different local optimums, and the growth speed of accumulated expected regret in all models decreases at first and then converges, and the accumulated adoption rate increases gradually and then converges. All these models learn the preferences of advertisers to some extent to improve the performance of the recommender system, compared with the recommendation strategy without a learning module.

To verify the generalization ability of the model, we conduct a controlled experiment, in which the experimental group is a model with a Dropout ratio of 40%, and the control group is the same

model without the preference-related information (only historical adoption information can be observed by the model). We draw the results in Figure 5, and the experimental results show that the performance of the model with the preference-related information is better than the model without it in both accumulated expected regret and accumulated adoption rate. This indicates that the model can learn the preferences of advertisers through understanding information related to $\mathbf{w}$, and thus verifying the generalization performance of the model.

6 CONCLUSION

In this work, we have investigated the problem of advertising strategy recommendations for online advertising. Through an online A/B test on a prototype recommender system, we demonstrate the potential benefits of recommending strategies to advertisers. We further augment the prototype system by recommending advertising strategy associated with the predicted advertising performance. We could exploit the advertisers' adoption behaviors to recommended strategies to learn advertiser's preference and optimize strategy adoption rate. We formulate the problem of strategy adoption rate maximization as a contextual bandit problem. The exploration is to learn the advertisers' preferences and optimization objectives, and the exploitation is to improve the adoption rate using the learned knowledge. We used Dropout trick to balance the exploration and exploitation. Experiments in a simulation environment verified the effectiveness of the system in the task of adoption rate optimization.

REFERENCES

[1] Justin Basilico and Thomas Hofmann. 2004. Unifying collaborative and content-based filtering. In *ICML*. 9.

[2] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. 2015. Weight uncertainty in neural networks. In *ICML*. 1613–1622.

[3] Han Cai, Kan Ren, Weinan Zhang, Kleanthis Malialis, Jun Wang, Yong Yu, and Defeng Guo. 2017. Real-time bidding by reinforcement learning in display advertising. In *WSDM*. 661–670.

[4] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & deep learning for recommender systems. In *DLRS*. 7–10.

[5] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *RecSys*. 191–198.

[6] David S Evans. 2008. The economics of the online advertising industry. *Review of Network Economics* 7, 3 (2008), 1–33.

[7] David S Evans. 2009. The online advertising industry: Economics, evolution, and privacy. *Journal of Economic Perspectives* 23, 3 (2009), 37–60.

[8] Y Gal and Z Ghahramani. 2016. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *ICML*. 1651–1660.

[9] Avi Goldfarb and Catherine Tucker. 2011. Online display advertising: Targeting and obtrusiveness. *Marketing Science* 30, 3 (2011), 389–404.

[10] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: a factorization-machine based neural network for CTR prediction. In *IJCAI*. 1725–1731.

[11] Liyi Guo, Rui Lu, Haoqi Zhang, Junqi Jin, Zhenzhe Zheng, Fan Wu, Jin Li, Haiyang Xu, Han Li, Wenkai Lu, et al. 2020. A Deep Prediction Network for Understanding Advertiser Intent and Satisfaction. In *CIKM*. 2501–2508.

[12] Xiaotian Hao, Zhaoqing Peng, Yi Ma, Guan Wang, Junqi Jin, Jianye Hao, Shan Chen, Rongquan Bai, Mingzhou Xie, Miao Xu, et al. 2020. Dynamic knapsack optimization towards efficient multi-channel sequential advertising. In *ICML*. 4060–4070.

[13] Eugene Ie, Vihan Jain, Jing Wang, Sanmit Narvekar, Ritesh Agarwal, Rui Wu, Heng-Tze Cheng, Tushar Chandra, and Craig Boutilier. 2019. SLATEQ: a tractable decomposition for reinforcement learning with recommendation sets. In *IJCAI*. 2592–2599.

[14] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[15] Brendan Kitts, Michael Krishnan, Ishadutta Yadav, Yongbo Zeng, Garrett Badeau, Andrew Potter, Sergey Tolkachov, Ethan Thornburg, and Satyanarayana Reddy Janga. 2017. Ad Serving with Multiple KPIs. In *SIGKDD*. 1853–1861.

[16] Volodymyr Kuleshov and Doina Precup. 2014. Algorithms for multi-armed bandit problems. *arXiv preprint arXiv:1402.6028* (2014).

[17] Sébastien Lahaie, David M Pennock, Amin Saberi, and Rakesh V Vohra. 2007. Sponsored search auctions. *Algorithmic Game Theory* 1 (2007), 699–716.

[18] Lihong Li, Wei Chu, John Langford, and Robert E Schapire. 2010. A contextual-bandit approach to personalized news article recommendation. In *WWW*. 661–670.

[19] Greg Linden, Brent Smith, and Jeremy York. 2003. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing* 7, 1 (2003), 76–80.

[20] Pasquale Lops, Marco De Gemmis, and Giovanni Semeraro. 2011. Content-based recommender systems: State of the art and trends. In *Recommender Systems Handbook*. 73–105.

[21] Jordan J Louviere, David A Hensher, and Joffre D Swait. 2000. *Stated choice methods: analysis and applications*. Cambridge University Press.

[22] Jie Lu, Dianshuang Wu, Mingsong Mao, Wei Wang, and Guangquan Zhang. 2015. Recommender system application developments: a survey. *Decision Support Systems* 74 (2015), 12–32.

[23] David JC MacKay. 1992. A practical Bayesian framework for backpropagation networks. *Neural Computation* 4, 3 (1992), 448–472.

[24] Takanori Maehara, Atsuhiro Narita, Jun Baba, and Takayuki Kawabata. 2018. Optimal bidding strategy for brand advertising. In *IJCAI*. 424–432.

[25] Radford M Neal. 1995. *Bayesian learning for neural networks*. Ph.D. Dissertation. University of Toronto.

[26] Seung-Taek Park, David Pennock, Omid Madani, Nathan Good, and Dennis DeCoste. 2006. Naïve filterbots for robust cold-start recommendations. In *SIGKDD*. 699–705.

[27] Michael J Pazzani and Daniel Billsus. 2007. Content-based recommendation systems. In *The Adaptive Web*. 325–341.

[28] Kan Ren, Weinan Zhang, Ke Chang, Yifei Rong, Yong Yu, and Jun Wang. 2017. Bidding machine: Learning to bid for directly optimizing profits in display advertising. *IEEE Transactions on Knowledge and Data Engineering* 30, 4 (2017), 645–659.

[29] Paul Resnick, Richard Zeckhauser, John Swanson, and Kate Lockwood. 2006. The value of reputation on eBay: A controlled experiment. *Experimental Economics* 9, 2 (2006), 79–101.

[30] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *WWW*. 285–295.

[31] J Ben Schafer, Dan Frankowski, Jon Herlocker, and Shilad Sen. 2007. Collaborative filtering recommender systems. In *The Adaptive Web*. 291–324.

[32] Jing-Cheng Shi, Yang Yu, Qing Da, Shi-Yong Chen, and An-Xiang Zeng. 2019. Virtual-taobao: Virtualizing real-world online retail environment for reinforcement learning. In *AAAI*. 4902–4909.

[33] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research* 15, 1 (2014), 1929–1958.

[34] William R Thompson. 1933. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika* 25, 3/4 (1933), 285–294.

[35] Di Wu, Xiujun Chen, Xun Yang, Hao Wang, Qing Tan, Xiaoxun Zhang, Jian Xu, and Kun Gai. 2018. Budget constrained bidding by model-free reinforcement learning in display advertising. In *CIKM*. 1443–1451.

[36] Xun Yang, Yasong Li, Hao Wang, Di Wu, Qing Tan, Jian Xu, and Kun Gai. 2019. Bid optimization by multivariable control in display advertising. In *SIGKDD*. 1966–1974.

[37] Weinan Zhang, Kan Ren, and Jun Wang. 2016. Optimal real-time bidding frameworks discussion. *arXiv preprint arXiv:1602.01007* (2016).

[38] Weinan Zhang, Yifei Rong, Jun Wang, Tianchi Zhu, and Xiaofan Wang. 2016. Feedback control of real-time display advertising. In *WSDM*. 407–416.

[39] Weinan Zhang, Shuai Yuan, and Jun Wang. 2014. Optimal real-time bidding for display advertising. In *SIGKDD*. 1077–1086.

[40] Weinan Zhang, Ying Zhang, Bin Gao, Yong Yu, Xiaojie Yuan, and Tie-Yan Liu. 2012. Joint optimization of bid and budget allocation in sponsored search. In *SIGKDD*. 1177–1185.

[41] Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep interest network for click-through rate prediction. In *SIGKDD*. 1059–1068.

[42] Han Zhu, Junqi Jin, Chang Tan, Fei Pan, Yifan Zeng, Han Li, and Kun Gai. 2017. Optimized cost per click in taobao display advertising. In *SIGKDD*. 2191–2200.

[43] Han Zhu, Xiang Li, Pengye Zhang, Guozheng Li, Jie He, Han Li, and Kun Gai. 2018. Learning tree-based deep model for recommender systems. In *SIGKDD*. 1079–1088.