Value-aware Recommendation based on Reinforcement Profit Maximization

Changhua Pei*
Alibaba Group
changhua.pch@alibaba-inc.com

Xinru Yang*◇
Carnegie Mellon University
xyang@andrew.cmu.edu

Qing Cui
Alibaba Group
cuiqing.cq@alibaba-inc.com

Xiao Lin
Alibaba Group
hc.lx@alibaba-inc.com

Fei Sun
Alibaba Group
ofey.sunfei@gmail.com

Peng Jiang
Alibaba Group
jiangpeng.jp@alibaba-inc.com

Wenwu Ou
Alibaba Group
santong.oww@taobao.com

Yongfeng Zhang§
Rutgers University
yongfeng.zhang@rutgers.edu

ABSTRACT

Existing recommendation algorithms mostly focus on optimizing traditional recommendation measures, such as the accuracy of rating prediction in terms of RMSE or the quality of top-k recommendation lists in terms of precision, recall, MAP, etc. However, an important expectation for commercial recommendation systems is to improve the final revenue/profit of the system. Traditional recommendation targets such as rating prediction and top-k recommendation are not directly related to this goal.

In this work, we blend the fundamental concepts in online advertising and micro-economics into personalized recommendation for profit maximization. Specifically, we propose value-aware recommendation based on reinforcement learning, which directly optimizes the economic value of candidate items to generate the recommendation list. In particular, we generalize the basic concept of click conversion rate (CVR) in computational advertising into the conversation rate of an arbitrary user action (XVR) in E-commerce, where the user actions can be clicking, adding to cart, adding to wishlist, etc. In this way, each type of user action is mapped to its monetized economic value. Economic values of different user actions are further integrated as the reward of a ranking list, and reinforcement learning is used to optimize the recommendation list for the maximum total value. Experimental results in both offline benchmarks and online commercial systems verified the improved performance of our framework, in terms of both traditional top-k ranking tasks and the economic profits of the system.

CCS CONCEPTS

• **Information systems → Recommender systems**.

*Equal contribution. Listing order is random.
◇This work was done when Xinru Yang was an intern at Alibaba.
§Corresponding author.

WWW '19, May 13–17, 2019, San Francisco, CA, USA

ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313404

KEYWORDS

Recommender Systems; Reinforcement Learning; Economics of Data Science

ACM Reference Format:
Changhua Pei, Xinru Yang*, Qing Cui, Xiao Lin, Fei Sun, Peng Jiang, Wenwu Ou, and Yongfeng Zhang. 2019. Value-aware Recommendation based on Reinforcement Profit Maximization. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313404

1 INTRODUCTION

Recommender system has become a fundamental service in many online applications. Years of research have witnessed the great advancement on the development of recommendation systems, and various different techniques have been proposed to support better performance in practice, including but not limited to content-based methods [23], user/item-based collaborative filtering methods [8], matrix factorization techniques [17], and the more recent deep learning-based approaches [33]. Most of the existing methods focus on two of the most fundamental tasks, namely, rating prediction and top-k recommendation. They usually aim at the optimization of rating or ranking-related measures, such as root mean square error (RMSE), mean average precision (MAP), and many others.

However, an important and sometimes the ultimate goal of commercial recommendation system is to gain revenue and profits for the platform, but traditional recommendation research are not directly related to this goal. They mostly focus on whether or not the algorithm can predict accurate ratings (rating prediction), or if the system can rank the clicked items correctly (top-k recommendation). Nevertheless, researchers have shown that improved performance on rating prediction does not necessarily improve the top-k recommendation performance in terms of user purchase [5]. And even better performance on top-k recommendation does not necessarily improve the profit of the system [34], because the purchased recommendations may not convert into the best profit due to their difference in price. For example, by recommending daily necessities, the system has a better chance of getting the user to purchase the recommendation, but the expected profit achieved from the purchase may be smaller than if a luxury good was purchased, though the latter has a smaller chance of being purchased.

In this work, we propose value-aware recommendation for profit maximization, which directly optimizes the expected profit of a recommendation list. The basic idea is to balance the unit profit of a recommendation and the chance that the recommendation is purchased, so that we can find the recommendation list of the maximum expected profit. To achieve this goal, we generalize the key concept of click conversion rates (CVR) [1, 19] in online advertising to the conversion rate of arbitrary user actions (XVR), which scales each user action into the monetized profit of the system based on large-scale user behavior logs.

Once the user actions are converted into expected profits, we further design an aggregated reward function based on the monetized user actions, and then adopt reinforcement learning to optimize the recommendation list towards the maximized expected profit for the system. For training, we developed a benchmark dataset by collecting the user behaviors in a real-world E-commerce system. Furthermore, we conducted online experiments on this E-commerce system to validate the online performance of value-aware recommendation in terms of both ranking performance and system profits.

The key contributions of the paper can be summarized as follows:

- We propose *value-aware recommendation* to maximize the profit of a recommendation system directly, which to the best of our knowledge, is the first time to explicitly optimize the profit of the personalized recommendation list in large-scale online system.
- We propose to generalize the basic concept of click conversation rate into the conversion rate of arbitrary user actions, so that each user action can be monetized into the profit of the system.
- By aggregating the monetized user actions into a unified profit reward, we develop a reinforcement learning-based algorithm for value-aware recommendation.
- We conduct both offline experiments with benchmark datasets and online experiments with real-world commercial users to validate the performance of value-aware recommendation on ranking and profit maximization.

2 RELATED WORK

In this section, we introduce the key related work in terms of three perspectives: economic recommendation, computational advertising, and reinforcement learning.

2.1 Economic Recommendation

For a long time, recommendation system research has been working on rating- or ranking-related tasks such as rating prediction, top-k recommendation, sequential recommendation, etc., and a lot of successful models have been proposed, which greatly advanced the performance of recommendation systems. To name a few, this includes content-based methods [23], user/item-based collaborative filtering [8], shallow latent factor models such as matrix factorization [17, 18, 22, 24], and more recent deep learning based methods [14, 31, 33]. However, the related methods seldom consider the economic value/profit that a recommendation list brings about to the system, although this is one of the most important goals for real-world commercial recommender systems. Some recent research on economic recommendation has begun to take care of the economic value of recommendation[3, 7, 39]. For example, Zhang et al. [34] proposed to maximize social surplus for recommendation, Zhao et al. [35] proposed to learn the substitutional and complementary relations between items for utility maximization, and Ge et al. [10] further proposed to maximize the marginal utility per dollar for recommendation to benefit user preference. However, none of the above methods explicitly maximizes the expected profits of a recommendation list to generate recommendations.

2.2 Computational Advertising

Currently, computational advertising [6] has been playing a crucial role in online shopping platforms, rendering it essential to apply an intelligent advertising strategy that can maximize the profits by ranking numerous advertisements in the best order [2, 11, 12]. In this scenario, the goal of advertisers is to maximize clicks and conversions such as item purchase. Therefore, cost-per-click (CPC) and cost-per-action (CPA) models are often adopted to optimize the click-through-rate (CTR) and the conversion-rate (CVR), respectively. Although technically there are a lot of analogy between advertising and recommendation, there is not much work trying to bridge these two principles. This work is one of our first attempts to integrate the economic consideration in adversting into recommender systems for value-aware recommendation.

2.3 Reinforcement Learning

Reinforcement Learning (RL) aims at automatically learning an optimal strategy from the sequential interactions between the agent and the environment by trial-and-error [29]. Some existed works have made efforts on exploiting reinforcement learning for recommendation systems. For instance, conventional RL methods such as POMDP (Partially Observable Markov Decision Processes) [27] and Q-learning [30] estimate the transition probability and store the Q-value table in modeling. However, they soon become inefficient with the sharp increase in the number of items for recommendation. Meanwhile, other approaches have been proven useful with an effective utilization of deep reinforcement learning. For instance, [38] employs Actor-Critic framework to learn the optimal strategy by a online simulator. Furthermore, [37] considers both positive and negative feedback from users recent behaviors to help find optimal strategy. [9] uses the multi-agent reinforcement learning to optimize the multi-scenario ranking. Meanwhile, [36] adopts RL to recommend items on a 2-D page instead of showing one single item each time.

To mitigate the performance degradation due to high-variance and biased estimation of the reward, [4] provides a stratified random sampling and an approximate regretted reward to enhance the robustness of the model. Similarly, [16] introduces DPG-FBE algorithm to maintain an approximate model of the environment to perform reliable updates of value functions.

3 GENERALIZED CONVERSION RATE

In order to maximize the profit of recommender system, we need to measure the the value of different user actions on the platform. In this section, we introduce how to generalize the basic concept of click conversion rate to rescale each user action into monetized profit. Here we use Gross Merchandise Volume (GMV) to measure the profit of the system. GMV is a measure used in online retailing, which indicates the total sales dollar value of the merchandise sold

through a particular marketplace over a certain time frame[1]. We use GMV and profit interchangeably aftermentioned.

3.1 Conversion Rate of Clicks

Click conversion rate (CVR) is widely used in online advertising to estimate the potential profit of an advertising item to the advertising platform [21, 40]. It is usually defined as the result of dividing the number of conversions by the number of total clicks that can be tracked to a conversion during the same period of time. In E-commerce system, it is difficult to learn an optimal recommendation strategy by simply calculating the profits of transactions because purchasing an item is a relatively sparse action of users. Instead, users tend to click an item much more frequently. CVR provides a way to map the dense action of clicking into a scaled range of profits. For E-commerce system, CVR is the first step towards value-aware recommendation[28, 32]. Eq, 1 shows the total estimated GMV brought by click, which shows that CVR plays a crucial role in profit maximization,

$$E[GMV] = \sum_i I(click, i) \cdot CVR(i) \cdot price(i) \tag{1}$$

where price(i) is the unit price of item i, $I(click, i)$ is an indicator of the occurrence of click on item i. If item i is clicked, $I(click, i) = 1$, otherwise, 0.

3.2 Generalization to other Actions

In E-commerce platform, a series of behaviors of a user is often formed step by step with transforming probabilities. In Figure **??**, the general shopping process is simplified into four steps. First, the user sees something of interests and gets an impression on the item, then the user will click to see more details and to eventually add it to the cart. The user can also add it to the wishlist so he can add it to the cart later. Eventually, some items in the cart will be purchased together and contribute to final profits. Adding to wishlist and adding to cart reveal a strong desire for the item even if no transaction is made eventually. Therefore, these actions are also considered to contribute to the value.

Here we generalize the concept of CVR to XVR to map each type of user action x to its monetized economic value. XVR is defined as the possibility of transition from certain action x to purchase. Using XVR, we can estimate the total profits of different actions on platform. By generalizing Eq. 1, we get Eq. 2,

$$E[GMV] = \sum_{x,i} V(x, i) = \sum_{x,i} I(x, i) \cdot XVR(i) \cdot price(i) \tag{2}$$

where E[GMV] denotes the expected GMV of the platform, x represents certain action including *click*, *add to cart*, *add to wishlist*, and *purchase*. Here $V(x, i)$=$I(x, i) \cdot XVR(i) \cdot price(i)$ is the expected profit given user has an action x.

4 REINFORCED PROFIT OPTIMIZATION

The main goal of this work is to maximize the total profit of recommender system in an E-commerce scenario. Here the profit indicate a total sales dollar value for merchandise sold online in a certain time frame. Using the generalized conversation rate, we can transfer all user actions to monetized profit and maximize it. Reinforcement learning (RL) aims to learn a policy that maximizes the accumulated reward. The profit can naturally be used as the reward with which we can find better actions (ranking policy). Besides, the view of interact between the environment (users) and RL agent fits our recommendation scenario well. Therefore, in this paper, we choose reinforcement learning, evolution strategy more specifically[25], as our algorithm for its simplicity and effectiveness.

4.1 Reinforcement Learning Modeling

Figure 2 illustrates how our value-based RL agent interacts with the user to maximize the profit. First, the user comes to the system at some time and makes an initial request, the system receives some information about the user as well as the candidate items as a state and responds to the request, which virtually means taking an action by generating an order to rank items. Next, the user shows a certain behaviors regarding the items such as click, add to wishlist, add to cart or purchase, which is used to calculate the reward to the system for what kind of action it takes.

In our recommendation platform, items are shown in cascade on a mobile App one by one. Each time the user initiates a request, 50 items are recommended to him/her. As user scrolls down the list and have seen all 50 items, a new request is triggered. This process is repeated until the user leaves the App or return the top of the cascade, labelled as "exit" in Figure 2. We use a metric called "pageid" to distinguish different requests in this interaction, similar to the concept of "page" to a search engine. As the user and the system interact with each other, the system learns how to respond to the state to obtain an optimized accumulative reward.

States: States are used to describe an user's request, including user-side, item-side, and contextual features. The user-side features include *age*, *gender*, *purchase power*. The item-side features are *ctr*, *cvr*, *price*. The contextual features include *pageid*, *request time*. In total, the states in our model can be formulated as <*age*, *gender*, *purchase power*, ctr_0, ..., ctr_{49}, cvr_0, ..., cvr_{49}, $price_0$, ..., $price_{49}$, *pageid*, *request time*>, the subscript i denotes the corresponding features of i^{th} item in the corresponding page.

All these features work together to influence the final GMV. For example, purchase power is calculated by how many and how expensive the bought items are. The higher the level of purchase power is, the more money is spent on the platform. Fig. 3a, 3b, and 3c show average GMV per user, number of users and summed GMV under different purchasing power and degree of activity. Degree of activity indicates the time a user spends on the platform. Fig. 3a shows users with higher purchase power have higher average GMV. Thus the model tends to recommend items with higher price to user who has high purchase power. However, for summed GMV, users with middle level of purchase power contribute most because the large number of users, as shown in Fig. 3b.

Request time (labelled as hour in Fig. 4) and pageid are also features used in our model. Fig. 4 shows that the the first page whose pageid equal to 0 contributes most to the summed GMV, as user scrolls downwards, pages with larger pageid have smaller possibilities to have impression on users. Thus the model should recommend items with higher conversion rate in the first page. Besides, model will learn different weights at different hour as Fig. 4 show the summed GMV varies with request time.

[1]https://en.wikipedia.org/wiki/Gross_merchandise_volume

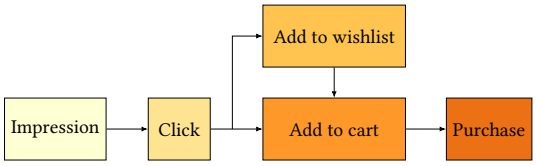


Figure 1: A flow of the user behavior series.

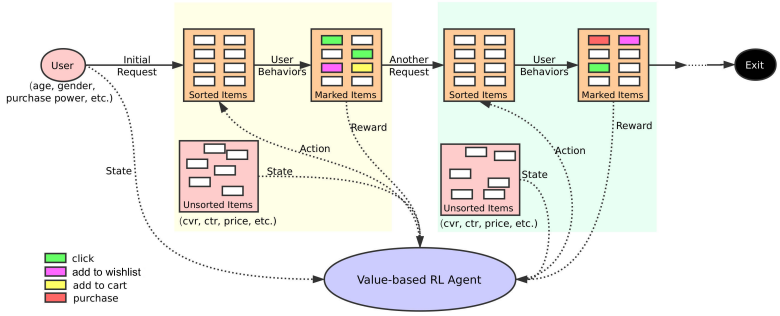
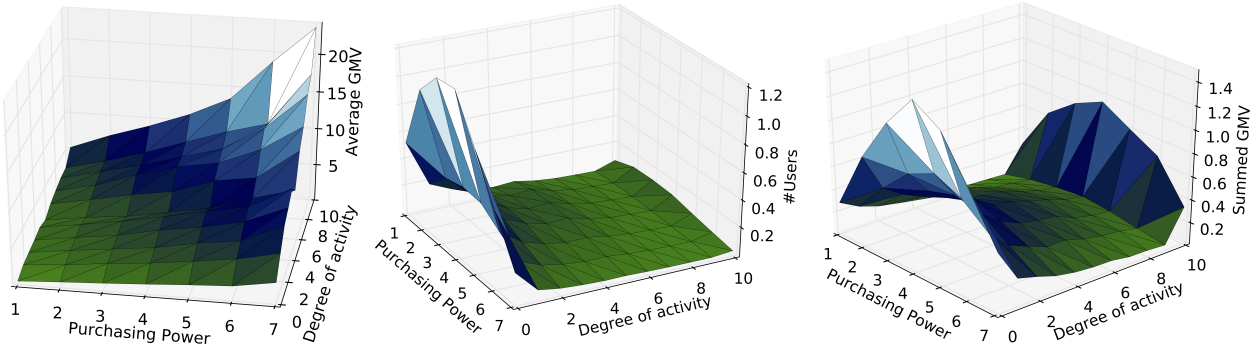


Figure 2: Structure of the model. When user initiates a request, the system will generate a recommendation list, and the user will interact with the list with certain actions. These Actions will be used to calculate the reward to update the RL Agent.



(a) Average GMV. **(b) Number of Users.** **(c) Summed GMV.**

Figure 3: Average/Summed GMV and the number of users under different level of purchasing power and degree of activity.

Actions: Action is the coefficient vector in the ranking formular which decides the order of candidate items. We design a ranking formula shown in Eq. 3 to order the items,

$$rankscore(i) = \sum_{x \in \mathcal{A}} P(x,i)^{\alpha_x} \cdot XVR(i)^{\beta_x} \cdot price(i)^{\gamma} \tag{3}$$

where $\mathcal{A}$ is the user action set which contains click, add to cart, add to wishlist. $P(x,i)$ is the possibility for a given user has action x on item i. $XVR(i)$ is the generalized conversion rate of item i. In total, the action can be formulated as $<\alpha_{click}, \alpha_{cart}, \alpha_{wishlist}, \beta_{click}, \beta_{cart}, \beta_{wishlist}, \gamma>$. If we only consider click in the action set $\mathcal{A}$, rankscore can be simplified to Eq. 4.

$$\begin{aligned} rankscore(i) &= P(click,i)^{\alpha_{click}} \cdot XVR^{\beta_{click}} \cdot price(i)^{\gamma} \\ &= ctr(i)^{\alpha_{click}} \cdot cvr(i)^{\beta_{click}} \cdot price(i)^{\gamma} \end{aligned} \tag{4}$$

Changing an action means to change the coefficient vector for the ranking formula.

Reward: In our value-based method, we use the expected profit as reward, which contains the monetized profit converted from all kinds of user actions. Based the definition of the expected GMV in Eq. 2, for a given item i, the reward R_i can be defined as:

$$R_i = V(click,i) + V(wishlist,i) + V(cart,i) + V(pay,i) \tag{5}$$

In this paper, unless specifically mentioned, we mainly use click and pay in reward, *i.e.*, $R_i = V(click,i) + V(pay,i)$. In the last section, we evaluate the performance when taking adding to cart and adding to wishlist into reward. The total reward of a recommendation list in a page that contains T items can be defined as $R_{page} = \sum_{i=0}^{T} R_i$.

4.2 Model Training

4.2.1 Offline Reward. In the above section, the reward of the model R_{page} is directly calculated with user's feedback online. This requires that the model is deployed online directly and learns the policy from the real-time data streaming. However, the online traffic is expensive. It is risky to let RL model learns directly online before we validate the effectiveness of the value-based method. To overcome this problem, we propose a simulated environment which leverage the historical data to approximate the feedback of the users. The simulated environment adopts a simple idea of NDCG (normalized discounted cumulative gain) that good policy

Table 1: Overview of the benchmark Dataset

clause	size(x10^6)
#distinct users	49
#distinct items	200
#requests	500
#clicks	670
#adding to carts	60
#adding to wishlists	30
#purchases	3

should rank the item user actually clicked and paid in the front. In this way, an offline reward R'_{page} is used to evaluate actions as following:

$$R'_{page} = \sum_{i=0}^{T} R_i * W_{\pi(i)} \tag{6}$$

For a given page with T items $1, \cdots, T$, the ranking policy can generate an order list, where $\pi(i) = k$ means item i is ranked at position k. We assign a discounted weight for each position k and higher position has greater weight, i.e. $W_{\pi(i)}$. Then we represent the reward as the weighted sum of item reward. In this paper, $W_{\pi(i)}$ is represented by exponential function which is $W_{\pi(i)} = \exp(-\pi(i))$.

4.2.2 Learning Algorithm. As the main goal of this paper is to evaluate the effectiveness of the value-aware recommendation, we choose the simplest model in reinforcement learning model family to exclude the influence of complex models. Finally we adopt evolution strategy algorithm (abbreviated ES algorithm). We train the model offline and deploy it online to perform A/B tests. The following introduces how the ES algorithm learns in the training process.

Our task is to learn a policy which maps state to action that maximizes the expected reward. To keep it simple, we use a linear model as the policy and θ is the parameters to be optimized. The ES algorithm starts from initial parameter θ_0 and then updates θ_t in the loop where t=1, 2, ... At each iteration loop t, the user's request for a new page and pass the state vector s_t to the model. The algorithm generates n baby-agent and each baby-agent j samples a set of increments ϵ_t^j from a normal distribution. Each model variant with parameter $\theta_t + \epsilon_t^j$ maps the state to an action. Each action orders the list in a different way and the user reacts to different ranking polices by clicking or purchasing certain items in the list. We calculate the reward of each policy j using Eq. 6 and labeled as $R'_{page}(j) = F(\theta_t + \epsilon_t^j)$. We collect all the rewards together and update θ_t eventually:

$$\theta_{t+1} \leftarrow \theta_t + \alpha_{lr} \frac{1}{n\sigma} \sum_{j=1}^{n} R'_{page}(j) \tag{7}$$

where σ and α_{lr} are hyper parameters, σ is the noise standard deviation and α_{lr} is the learning rate.

5 EXPERIMENTS

In this section, we first introduce the settings of our benchmark dataset used for training or offline evaluation. Then we evaluate our method both online and offline together with baselines. [2]

[2]The code and dataset of the paper are released at https://github.com/rec-agent/rec-rl.

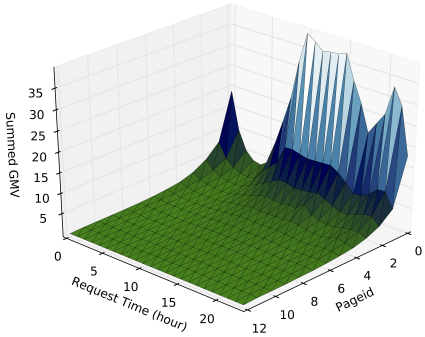


Figure 4: GMV across different pages and time.

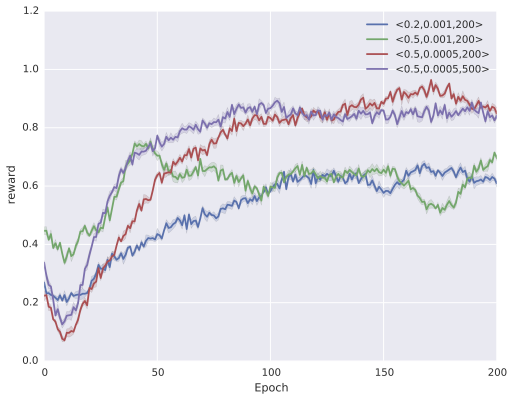


Figure 5: The training curve under different parameters: <noise standard deviation, learning rate, batch size>

5.1 Benchmark Dataset

Our benchmark dataset is collected from real-world E-commerce platform and can simulate the interactive environment and users' behaviors. Each piece of data is comprised of two parts: the *states* and users' feedback: click, add to cart, add to wishlist, purchase. Table 1 lists the details of the dataset.

5.2 Experimental Setup

5.2.1 ***Baselines****:* We use the following three methods as baselines to compare with our value-based RL method: Item-based Collaborative Filtering, LR-based learning to rank(LTR), DNN-based learning to rank.

- **Item-based CF**: The classic Item-based CF [26] is widely used for recommendation. compared to user-based CF [15], Item-based CF is more scalable and accurate in E-commerce recommendation because the relationships between items are more stable. In this paper, we use a fine-tuned item-based collaborative filtering method as one of our baselines. Other CF methods such as matrix factorization can also be used as baselines. However, as they are all non value-aware methods, we omit them in this paper for simplicity.
- **LR-based LTR**: The Point-wise Learning-To-Rank paradigm [20] is adopted to learn the rankings of recommended items with Logistic Regression(LR) as the ranking model. We use clicked and purchased items as positive samples to learn a binary classification model, and the price is used as the weight of loss. To reduce the variance of price and make the optimization stable, we actually use $\log(price)$ as the weight. For fair comparison, here

Table 2: Offline Evaluation Results (p-value < 0.005). Numbers in the brackets are improvements compared to Item-based CF and LR-based LTR.

	E[GMV]	Average R'_{page}	Precision@20(%)	Recall@20(%)	MAP@20(%)
Item-based CF	0.40	19.73	2.96	27.93	8.69
LR-based LTR	0.49 (22.5%/-)	25.87 (31.1%/-)	3.45 (16.6%/-)	32.42 (16.1%/-)	11.04 (27.0%/-)
DNN-based LTR	0.50 (25.0%/2.0%)	25.88 (31.2%/0.04%)	3.65 (23.3%/5.8%)	34.01 (21.8%/4.9%)	12.27 (41.2%/11.1%)
Value-based RL	0.53 (32.5%/8.2%)	27.78 (40.8%/7.4%)	3.74 (26.4%/8.4%)	34.82 (24.7%/7.4%)	12.36 (42.2%/12.0%)

Table 3: Online Evaluation Results (p-value < 0.005).

	GMV	CTR(%)	IPV
Item-based CF	7.57	3.04	2.48
LR-based LTR	8.95 (18.3%/-)	3.26 (7.4%/-)	2.67 (7.5%/-)
DNN-based LTR	9.06 (19.7%/1.2%)	3.28 (8.1%/0.6%)	2.69 (8.3%/0.7%)
Value-based RL	9.68 (27.9%/8.2%)	3.29 (8.2%/0.9%)	2.70 (8.8%/1.1%)

we employ the same feature set as our proposed value-based RL. There are three hyper-parameters involved in the training of LR-based LTR, *i.e.*, the ℓ_2 regularization weight, the learning rate and the batch size. These parameters are set to 0, 0.5 and 128 after tuning. We also test a repression model for predicting the profit. However, it cannot beat classification models like LR. This is consistent with previous work [13], regression model may not be suitable for ranking problems in recommender system.

- **DNN-based LTR**: This method is the same as LR-based LTR, except that the ranking model is a Deep Neural Network (DNN) instead. The neural network involves two hidden layers with 32 and 16 neurons respectively. The ℓ_2 regularization weight, the learning rate and the batch size are set to 0, 0.05 and 1024. We also tested network structures including RNN, attention networks, and Wide&Deep network. Optimizer, activation functions, normalization, and other hyper parameters are tuned by greedy search. However, the improvement of these networks is not significant on very large dataset.

5.2.2 ***Value-based RL****:.* Different hyper parameters are tried by a greedy search to find the better training performance. In ES algorithm, under the same training set, the larger reward a model can achieve, the better the model is. We name some important parameters in Fig. 5 to illustrate this. Having a larger deviation (from 0.2 to 0.5) can greatly improve the maximum reward the model can achieve. However, the model is not stable so that the reward will drop after some iterations. We adjust the learning rate in Adam optimizer from 0.001 to 0.0005 and the training curve become stable. We also try larger batch(from 200 to 500), the improvement is not that much. Finally, <0.5,0.0005,200> are used as noise standard deviation, learning rate and batch size in our RL model respectively.

5.3 Key Results

Offline Evaluation: In offline evaluation, we train the model on the data of one week and evaluate it on the data of the following week. Precision, recall and MAP within the top 20 items is used to measure the performance. Besides the traditional metrics, average R'_{page} (Eq. 6) and E[GMV] (Eq. 2) are used as value-based metrics.

Table 2 shows that the Learning-To-Rank (LTR) baseline outperforms item-based CF in all measures which indicates the LTR methods can predict the profit more effectively. We leverage price in two ways. On one hand, LTR methods use price as one of the features. On the other hand, LTR methods use price as the weight for loss in the learning process. Our value-based approach has 6.0% and 7.3% improvements in E[GMV] and R'_{page} respectively when compared to DNN-based LTR. Besides, Value-based RL achieves better performance in precision(2.5%), recall(2.4%) and MAP(0.7%) than DNN-based LTR with same features. Our approach can precisely monetize an arbitrary user action into the profit by generalizing the basic concept of click conversion rate (CVR) to CXR.

Table 4: Offline performance when considering adding to cart and adding to wishlist (p-value < 0.005).

	E[GMV]	Precision @20(%)	Recall @20(%)	MAP (%)
click	42.2	3.67	34.5	12.1
click,cart,wishlist	43.5	3.88	35.8	12.9
	(3.1%)	(5.7%)	(3.8%)	(6.6%)

Online Evaluation: An online A/B test is used to evaluate different methods. Metrics we use in A/B test are ctr, IPV and GMV. IPV is the absolute number of clicked items on the platform, which acts a supplement metric to ctr. To make the online evaluation comparable, each bucket in A/B test has the same number of users and contains over millions of users.

Table 3[3] shows the average results for one week. Online evaluation shows results are consistent with the offline evaluation. The DNN-based LTR improves both GMV, ctr and IPV by 19.7%, 8.1% and 8.3% respectively compared to Item-based CF. Even though the DNN-based LTR does not explicitly optimizing the GMV, it has 19.7% improvements in GMV compared to item-based CF. By explicitly mapping the value of different actions into the GMV, our value-based RL method can bring another 6.8% improvement on GMV compared to DNN-based LTR. The improvements on ctr and IPV is 0.3% and 0.4% respectively. Online test is done in real-world system (involving 2 hundred million clicks per day, 7 days). Based on significance test (p<0.005), the improvement are all significant and have business values for E-commercial platform, because every thousandth of improvement on ctr means hundreds of millions of extra clicks in real-world systems.

5.4 Click vs. Add to Cart and Add to Wishlist

In this part, we take adding to cart and adding to wishlist into consideration to evaluate the performance of our value-based model. Results in Table 4 show offline performance when considering the value of adding to cart and adding to wishlist actions into the model. The improvement on expected GMV is 3.1%. The improvements on precision, recall and MAP are 5.7%, 3.8% and 6.6% respectively. It

[3]Normalized related value instead of the original absolute value of the online measures are shown for business reason. The relative improvement is consistent with real experimental results.

means adding to cart and adding to wishlist actions are useful for the value-based profit maximization.

6 CONCLUSIONS AND FUTURE WORK

Great advances have been achieved by existing research to improve the accuracy of rating prediction and the quality of top-k recommendation lists. However, the economic value of recommendation is rarely studied. For large-scale commercial recommendation system, state-of-the-art algorithms seldom maximize the final revenue/profit of the system. To eliminate this gap, we propose value-aware recommendation to maximize the profit of commercial recommendation system directly. Specifically, we generalize the basic concept of click conversion rate (CVR) to the conversation rate of an arbitrary user action on the platform (XVR), where different actions of users(*click, add to cart, add to wishlist, purchase*)can be monetized into the profit of the system. Then we use reinforcement learning (RL) to maximize the profit whose reward is the aggregated monetized user actions. Both offline and online experiments show that our value-based RL model not only performs better on traditional metrics such as precision, recall and MAP, but also greatly improves the final profit than existing methods. This paper acts as the first step towards value-aware recommendation and further improvement can be achieved by designing more powerful features and RL models in the future.

REFERENCES

[1] Ashish Agarwal, Kartik Hosanagar, and Michael D Smith. 2011. Location, location, location: An analysis of profitability of position in online advertising markets. *Journal of marketing research* 48, 6 (2011), 1057–1073.

[2] Andrei Broder, Marcus Fontoura, Vanja Josifovski, and Lance Riedel. 2007. A semantic approach to contextual advertising. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 559–566.

[3] Robin Burke, Gediminas Adomavicius, Ido Guy, Jan Krasnodebski, Luiz Pizzato, Yi Zhang, and Himan Abdollahpouri. 2017. VAMS 2017: Workshop on Value-Aware and Multistakeholder Recommendation. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*. ACM, 378–379.

[4] Shi-Yong Chen, Yang Yu, Qing Da, Jun Tan, Hai-Kuan Huang, and Hai-Hong Tang. 2018. Stabilizing reinforcement learning in dynamic environment with application to online recommendation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM.

[5] Paolo Cremonesi, Yehuda Koren, and Roberto Turrin. 2010. Performance of recommender algorithms on top-n recommendation tasks. In *Proceedings of the fourth ACM conference on Recommender systems*. ACM, 39–46.

[6] Kushal Dave, Vasudeva Varma, et al. 2014. Computational advertising: Techniques for targeting relevant ads. *Foundations and Trends® in Information Retrieval* 8, 4–5 (2014), 263–418.

[7] Gediminas Adomavicius Dietmar Jannach. 2017. Price and Profit Awareness in Recommender Systems. *arXiv preprint arXiv:1801.00209* (2017).

[8] Michael D Ekstrand, John T Riedl, Joseph A Konstan, et al. 2011. Collaborative filtering recommender systems. *Foundations and Trends® in Human–Computer Interaction* 4, 2 (2011), 81–173.

[9] Jun Feng, Heng Li, Minlie Huang, Shichen Liu, Wenwu Ou, Zhirong Wang, and Xiaoyan Zhu. 2018. Learning to Collaborate: Multi-Scenario Ranking via Multi-Agent Reinforcement Learning. In *WWW*. 1939–1948.

[10] Yingqiang Ge, Shuyuan Xu, Shuchang Liu, Shijie Geng, Zuohui Fu, and Yongfeng Zhang. 2019. Maximizing Marginal Utility per Dollar for Economic Recommendation. In *WWW*.

[11] Anindya Ghose and Sha Yang. 2009. An empirical analysis of search engine advertising: Sponsored search in electronic markets. *Management science* 55, 10 (2009), 1605–1622.

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

[13] Asela Gunawardana and Guy Shani. 2009. A survey of accuracy evaluation metrics of recommendation tasks. *Journal of Machine Learning Research* 10, Dec (2009), 2935–2962.

[14] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 173–182.

[15] Jonathan L Herlocker, Joseph A Konstan, Al Borchers, and John Riedl. 2017. An algorithmic framework for performing collaborative filtering. In *ACM SIGIR Forum*, Vol. 51. ACM, 227–234.

[16] 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 *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM.

[17] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 8 (2009), 30–37.

[18] Daniel D Lee and H Sebastian Seung. 2001. Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems*. 556–562.

[19] Kuang-chih Lee, Burkay Orten, Ali Dasdan, and Wentong Li. 2012. Estimating conversion rate in display advertising from past erformance data. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 768–776.

[20] Tie-Yan Liu. 2009. Learning to Rank for Information Retrieval. *Found. Trends Inf. Retr.* 3, 3 (March 2009), 225–331.

[21] Xiao Ma, Liqin Zhao, Guan Huang, Zhi Wang, Zelin Hu, Xiaoqiang Zhu, and Kun Gai. 2018. Entire Space Multi-Task Model: An Effective Approach for Estimating Post-Click Conversion Rate. In *Proceedings of the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, 1137–1140.

[22] Andriy Mnih and Ruslan R Salakhutdinov. 2008. Probabilistic matrix factorization. In *Advances in neural information processing systems*.

[23] Michael J Pazzani and Daniel Billsus. 2007. Content-based recommendation systems. In *The adaptive web*. Springer, 325–341.

[24] Steffen Rendle and Christoph Freudenthaler. 2014. Improving pairwise learning for item recommendation from implicit feedback. In *Proceedings of the 7th ACM international conference on Web search and data mining*. ACM, 273–282.

[25] Tim Salimans, Jonathan Ho, Xi Chen, Szymon Sidor, and Ilya Sutskever. 2017. Evolution strategies as a scalable alternative to reinforcement learning. *arXiv preprint arXiv:1703.03864* (2017).

[26] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*. ACM, 285–295.

[27] Guy Shani, David Heckerman, and Ronen I Brafman. 2005. An MDP-based recommender system. *Journal of Machine Learning Research* 6, Sep (2005), 1265–1295.

[28] SIGKDD. 2015. Life-stage Prediction for Product Recommendation in E-commerce. ACM, 1879–1888.

[29] Richard S Sutton, Andrew G Barto, et al. 1998. *Reinforcement learning: An introduction*. MIT press.

[30] Nima Taghipour and Ahmad Kardan. 2008. A hybrid web recommender system based on q-learning. In *Proceedings of the 2008 ACM symposium on Applied computing*. ACM, 1164–1168.

[31] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. 2015. Collaborative deep learning for recommender systems. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1235–1244.

[32] Qiaolin Xia, Peng Jiang, Fei Sun, Yi Zhang, Xiaobo Wang, and Zhifang Sui. 2018. Modeling Consumer Buying Decision for Recommendation Based on Multi-Task Deep Learning. In *CIKM*. ACM, 1703–1706.

[33] Shuai Zhang, Lina Yao, and Aixin Sun. 2017. Deep learning based recommender system: A survey and new perspectives. *arXiv preprint arXiv:1707.07435* (2017).

[34] Yongfeng Zhang, Qi Zhao, Yi Zhang, Daniel Friedman, Min Zhang, Yiqun Liu, and Shaoping Ma. 2016. Economic recommendation with surplus maximization. In *WWW*.

[35] Qi Zhao, Yongfeng Zhang, Yi Zhang, and Daniel Friedman. 2017. Multi-product utility maximization for economic recommendation. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. ACM, 435–443.

[36] Xiangyu Zhao, Long Xia, Liang Zhang, Zhuoye Ding, Dawei Yin, and Jiliang Tang. 2018. Deep Reinforcement Learning for Page-wise Recommendations. In *Proceedings of the 12th ACM Conference on Recommender Systems*. ACM.

[37] Xiangyu Zhao, Liang Zhang, Zhuoye Ding, Long Xia, Jiliang Tang, and Dawei Yin. 2018. Recommendation with Negative Feedback via Pairwise Deep Reinforcement Learning. *Proceedings of the 24th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (2018).

[38] Xiangyu Zhao, Liang Zhang, Zhuoye Ding, Dawei Yin, Yihong Zhao, and Jiliang Tang. 2017. Deep Reinforcement Learning for List-wise Recommendations. *arXiv preprint arXiv:1801.00209* (2017).

[39] Yong Zheng. 2017. Multi-Stakeholder Recommendation: Applications and Challenges. *arXiv preprint arXiv:1707.08913* (2017).

[40] Han Zhu, Junqi Jin, Chang Tan, Fei Pan, Yifan Zeng, Han Li, and Kun Gai. [n. d.]. Optimized cost per click in taobao display advertising.