An Intelligent Bidding Strategy Based on Model-Free Reinforcement Learning for Real-Time Bidding in Display Advertising

Mengjuan Liu, Jiaxing Li, Wei Yue, Lizhou Qiu, Jinyu Liu, Zhiguang Qin
School of Information and Software Engineering, University of Electronic Science and Technology of China
Chengdu, China
mjliu@uestc.edu.cn

***Abstract*—In recent years, the most important paradigm in online display advertising is real-time bidding (RTB). It allows advertisers to buy individual ad impressions through real-time auctions, to obtain maximum revenue. However, the existing strategies usually bid an ad impression independently, ignoring the impacts of each bid on the overall revenue during the whole ad delivery period. Thus, the recent research suggests that using the reinforcement learning (RL) framework to learn the optimal bidding strategy in RTB, based on both the immediate and future rewards. In this paper, we formulate budget constrained bidding as a model-free reinforcement learning problem, where the state space is presented by the impressions' feature parameters and the auction information, while an action is to set the bidding price. Different from the prior value-based model-free work, which suffers from the convergence problem, we learn the optimal bidding strategy by employing the policy gradient model. Additionally, we design four reward functions according to different auction results and user feedback to the learned bidding strategy more in line with the optimization objectives. We evaluate the performance of the proposed bidding strategy based on a real-world dataset, and the experimental results have demonstrated the superior performance and high efficiency compared to state-of-the-art methods.**

Keywords*—Real-Time Bidding, Bidding Strategy, Reinforcement Learning

I. INTRODUCTION

Real-time Bidding (RTB) [1,2] has been the most important paradigm in online display advertising since 2011, which provides an open and transparent channel for publishers and advertisers to sell and buy ad impressions automatically in real-time. As a new business model of the online advertising market, RTB is different from traditional sponsored search or contextual advertising. It allows advertisers to set a bid price for each ad impression in real-time. In RTB display advertising, there are usually four platforms, namely ad exchange (ADX), demand-side platform (DSP), supply-side platform (SSP) and data management platform (DMP), shown as in Fig.1. Here, the ADX combines multiple ad networks together and holds auctions to determine which campaigns win the impressions; the DSP serves advertisers by helping manage their budgets and bid for each impression; the SSP works as the agency of publishers by selling impressions; and the DMP collects user data and sells it anonymously to the DSP, SSP, ADX and sometimes to advertisers directly in real-time bidding (RTB) for better matching between ads and users [3].

In Fig.1, we present the process of how an impression is auctioned. In detail, an ad impression is created when a user visits a webpage, and an ad request will be immediately sent by the SSP to the ADX. Meanwhile, a bid request is published by the ADX for the connected DSPs. Then the DSPs generate different bid prices based on the user features generating from the DMPs and webpage information corresponding to this bid request to the ADX. After that, the ADX determines the winner according to the generalized second pricing (GSP) mechanism [4]. Finally, the winner pays for the impression and its ad will be displayed to the online user through the webpage (app). The entire process will be completed in 10-100 milliseconds. More details of RTB are given in [5].

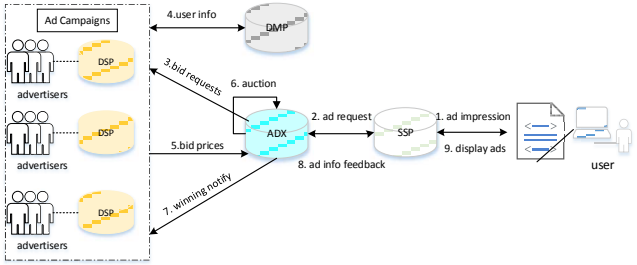

Fig. 1. The general process of an ad delivery in RTB

In RTB, the DSP is treated as an agent of advertisers. It helps the advertisers spend the ad campaigns' budgets on the most high-quality ad impressions to obtain more positive user feedback, such as clicks or conversions [19]. Thus, we consider that one of the fundamental problems is to capture those valuable ad impressions in real-time. The solution to this problem is called as ***a bidding strategy*** in the DSP, which calculate the bid price for each impression to enable advertisers to achieve their optimization goals. The existing bidding strategies are based on the evaluation of the impression to the ad campaign, such as the predicted click-through rate (pCTR) and the predicted conversion rate (pCVR). There is a common issue that they consider the bid decision as a static optimization problem of treating the value of each impression independently. However, such a static optimal strategy may not work well because of the dynamic characteristics of the RTB market. For example, the DSP cannot predict whether the impressions which may lead to the positive feedback will arrive before the budget runs out. As such, each bid is strategically correlated by the constrained budget and the overall effectiveness of the campaign (e.g., the rewards from generated clicks), and the dynamics of the RTB environment [5]. Fortunately, reinforcement learning (RL) is a promising solution for the above problem. By RL, we model the bidding strategy as a sequentially dynamic interactive process, so that the budget of a campaign can be dynamically allocated across all the available impressions based on both the immediate and long-term future rewards.

In this paper, we propose an intelligent bidding strategy by leveraging policy-based model-free reinforcement learning framework. In our approach, a bidding agent is designed to

978-1-7281-5141-0/19/$31.00 ©2019 IEEE
DOI 10.1109/CBD.2019.00051

decide the bid prices sequentially for each matched impression. Pointedly, when the agent receives a bid request, it first observes the current state and then generates the corresponding action (bid price) by a deep neural network (DNN). After an auction, the agent will obtain different rewards from the environment according to different auction results and user feedback. Here, the ad campaign's information (such as the remaining budget and the life span) and the impression's feature vectors are regarded as the state; the action is defined as a bid price; and the reward takes into account both cost and benefit. At last, we adopt the policy gradient algorithm to maximize the cumulative revenue of an ad campaign to approximate the optimal bidding strategy. Our method is quite different from the existing RL-based bidding strategies, most of which adopt value-based methods (such as Q-Learning [15], Sarsa [17], and Deep Q Network [16]). The main contributions of this paper are summarized as follows:

- We present a policy-based model-free RL approach to learn the bidding strategy in RTB for the first time, which significantly improves the convergence and efficiency, compared with other approaches adopting value-based RL;
- We design four reward functions according to the different auction results and user feedback, which guide the bidding agent to learn the optimal bidding strategy under budget constraints;
- We conduct the experiments on a public dataset, and the results verify the effectiveness of the proposed RL framework for learning the optimal bidding policy.

The rest of this paper is organized as follows. Section II introduces related work. Section III and IV define and detail our proposed RL-based bidding strategy. The experimental setting and results are presented and analyzed in Section V. Finally, we conclude our work in Section VI.

II. RELATED WORK

As mentioned in Section I, the bidding strategy is crucial for advertisers to maximize their budget cost efficiency of each ad campaign. The authors in [4] proved that truthful bidding would obtain the optimal revenue in the second-price auctions; but it is hard to implement in a real auction environment. The linear bidding [6] is widely practiced in many real-world RTB systems. However, it cannot maximize the revenue of an ad campaign by the linear bidding strategy, so the authors in [7] proposed an optimal bidding strategy with the non-linear form which is better than the linear one under variant budget constraints. Unfortunately, both the linear and the non-linear bidding strategies, optimize the bidding as a static process and bid for each impression independently. They set the bid price for each impression only based on its evaluation (usually, pCTR), not considering the competition degree of the auction market and the available budget in the new ad delivery period.

In [5], the bidding decision process is formulated as a Markov Decision Process (MDP) and generates a bid price by a dynamic programming algorithm. However, the model-based RL approaches such as [5] require the explicit state transition matrix; but in practice, the arrival of impressions is episodic, so the matrix is difficult to represent mathematically. So the RTB should be considered as a model-free environment. The authors in [8] proposed a new approach for budget-constrained bidding by leveraging model-free RL, where the bidding problem is formulated as a parameter control problem based on the linear bidding equation. However, the value-based model-free RL algorithms may have convergence issues in practice and all have shown unable to converge to any policy for simple MPDs and simple function approximator [9,10]. Besides, the optimal policy is to select different actions with specific probabilities rather than deterministic policies that output by the value-based approach [11]. On the other hand, policy gradient methods can theoretically guarantee better convergence properties and keep also effective in high-dimensional or continuous action spaces [11]. In this paper, we adopt a policy gradient approach that is different from [5, 8] to solve the problem that the value-based RL cannot easily converge to the optimal strategy. Also, we do many experiments under different budget constraints and the results demonstrate the effectiveness of our method.

III. PROBLEM DEFINITION

In this section, we will formulate the problem of real-time bidding strategy in the reinforcement learning framework and describe the four reward functions that we propose in detail.

A. Bidding based on Reinforcement Learning

Reinforcement learning is an effective solution to the finite Markov Decision Process (S, A, P, R). S and A represent states and actions respectively, and the state transition matrix and reward are given by P and R. The interaction process between the bidding agent and environment is summarized as follows: the agent observes state s_t from the environment and takes action a_t according to strategy π in each discrete time step $t \in 1, 2, ..., \mathrm{T}$ (where an impression corresponds to a time step in this paper), then the environment returns the reward r_t to the agent, and the agent observes the next state s_{t+1}. Because the time is limited, MDP will terminate after T time steps (or after the budget are exhausted). Finally, the goal of the agent is to maximize the cumulative discount reward, defined as

$$R = \sum_{t=1}^{T} \gamma^{t-1} r_t, \gamma \in [0,1]$$

where r_t is the immediate reward of state transition $s_t \to s_{t+1}$, and γ=1 because we consider bidding in RTB is an episodic episode. The overall process is shown in Fig. 2 below.

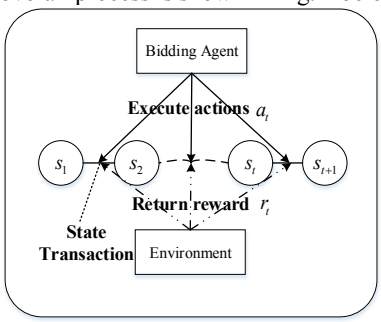

Fig. 2. Markov decision process

For our method, the main concepts are as follows：

State s : $s = < b, t, \theta, \overrightarrow{auct} >$ represents a state which contains the available ratio of the budget b , the remained ration of the expected number of impressions t in the ad delivery period, the predicted click-through rate θ of each impression, and $\overrightarrow{auct}$ is the impression's related feature vectors which generated by Factorization Machine (FM) model [12];

Action a : the agent's bid price for each ad impression, $a_t \in A = \{1, 2, 3..., 300\}$;

Reward r : immediate revenue from the transition $s_t \rightarrow s_{t+1}$, such as clicks or conversions. In our work, we get the revenue from four reward functions below;

Episode *ep* : in this paper, we treat an ad campaign's delivery period as an episode.

B. Reward Functions

Adhering to the principle of "reward for merit, punishment for error." We design four reward functions according to different auction results and user feedback: winning and losing impressions which may lead to clicks; winning and losing impressions which may not lead to clicks; so as to the learned bidding strategy more in line with the optimization goals. It is worth noting that our reward functions are updated in real-time on the impression level--the corresponding reward is generated according to the user feedback and the auction result of each impression.

Our reward functions are logically defined as (1):

$$reward_{win-clk} = E[r_t \mid s_t, a_t] \tag{1}$$

where the subscript of the reward is indicated by $win - clk$, win represents winning or not, and clk represents whether the impression is clicked.

1) Winning the impression that leads to a click: When the *t*-th impression arrives, (2) describes the reward degree of the agent for capturing the high-quality impression that leads to a click. Where m_t is the impression's market price, r_t is the revenue for the impression's true value [3], and we denote $pCTR_t$ the t-th impression's predicted click-through rate. CPC denotes cost per click. This reward function shows the positive feedback for the agent.

$$\begin{aligned} reward_{1-1} &= \left(1 - \left(\frac{a_t - m_t}{a_t}\right)^2\right) \times |r_t - m_t| \\ r_t &= pCTR_t \times CPC \end{aligned} \tag{2}$$

2) Winning the impression that may not be clicked: If the agent allocate the budget on the low-quality impression, it will be punished. Equation (3) shows the punishment for the agent. Where *punishWinRate* is the penalty factor, in the offline experiment, it is set to indicate the penalty degree of the agent for capturing worthless impressions according to the ratio of the remaining real clicks *remainClks* in the historical data and the available budget *remainBudget* in the episode.

$$\begin{aligned} reward_{1-0} &= -m_t * punishWinRate \\ punishWinRate &= \frac{remainClks}{remainBudgets} \end{aligned} \tag{3}$$

3) Losing the impression that leads to a click: If the agent loses the impression that may be clicked, it will get the punishment shown in (4), due to the loss of the potential revenue. Where *punishNoWinRate* is set to indicate the penalty degree of the agent for the ignoring of high-quality impressions with the value r_t . Moreover, it is related to the number of both not-winning impressions *withClkNoWinAucs* that maybe clicked and all impressions *withClkAucs* that lead to clicks, in real-time.

$$\begin{aligned} reward_{0-1} &= -\frac{r_t}{punishNoWinRate} \\ punishNoWinRate &= 1 - \frac{withClkNoWinAucs}{withClkAucs} \end{aligned} \tag{4}$$

4) Losing the impression that may not be clicked: If the agent loses the impression that not leading to a click, it will get the positive feedback from the reward function in (5).

$$\begin{aligned} reward_{0-0} &= \frac{baseEncourage}{encourageRate} \\ encourageRate &= 1 - \frac{withNoClkNoWinAucs}{withNoClkAucs} \end{aligned} \tag{5}$$

For the agent, allocating the budget to the low-quality ad impressions is ineffectively to obtain long-term revenue. Thus, we introduce $reward_{0-0}$ to guide the agent to learn the optimal strategy. Since the budget is not wasted for the low-quality impression, we take the market price m_t as the basic reward $baseEncourage$. The reward is described as $encourageRate$ according to the number of not-winning impressions *withNoClkNoWinAucs* that maybe not clicked and all impressions *withNoClkAucs* that not leading to clicks, in real-time.

IV. REINFORCEMTN LEARNING TO INTELLIGENTLY BID

In this section, we will introduce in detail the proposed Reinforcement Learning to Intelligently Bid framework, called *RLIB*. Our framework is built on top of REINFORCE [13], where the action selection policy is represented by a neural network whose input is a representation of the state s_t , whose output is an action selection policy $\pi(a_t | s_t; \theta)$, and whose weights are the policy parameters θ . In practice, a state-action sequence $\tau(s_1, a_1, s_2 ... s_T, a_T, s_{T+1})$ could be observed when the agent interacts with the episodic environment. And the agent's goal is to obtain an optimal policy π^* shown in (6) which can maximize the cumulative discount reward.

$$\pi^*(\tau;\theta) = \arg\max_{\theta} E_{\tau\sim\pi_\theta(\tau)}\left[\sum_{t=1}^{T}\gamma^{t-1}r_t\right] \tag{6}$$

Where $\pi(\tau;\theta)$ is the occurrence probability of the state action sequence τ. According to MDP, it is defined as (7):

$$\begin{aligned}\pi(\tau;\theta) &= p(s_1,a_1,...,s_T,a_T,s_{T+1};\theta)\\ &= p(s_1)\prod_{t=1}^{T}\pi(a_t|s_t;\theta)p(s_{t+1}|s_t,a_t)\end{aligned} \tag{7}$$

At each time step t, the agent observes the state s_t upon the probability distribution $p(s_t)$ and determines its bid price a_t, then it observes the next state s_{t+1} according to the state transition probability distribution $p(s_{t+1}\mid s_t,a_t)$. In general, we mark $\sum_{t=1}^{T}\gamma^{t-1}r_t$ as $r(\tau)$, which represents the cumulative discount reward for the state-action sequence and denote the optimal objective function as (8):

$$J(\theta) = E_{\tau\sim\pi(\tau;\theta)}[r(\tau)] = \int r(\tau)\pi(\tau;\theta)d\tau \tag{8}$$

Next, we consider how to change the probability distribution of the policy $\pi(a_t|s_t;\theta)$ by adjusting the parameters θ, resulting in the improvement of $r(\tau)$. Unfortunately, the explicit functions of $p(s_t)$ and $p(s_{t+1}\mid s_t,a_t)$ are typically unknown, due to the dynamics of the RTB environment. Here, we present a mathematical deduction shown in (9) for the gradient $\nabla J(\theta)$ of the optimal objective function with respect to the policy parameters θ. And it does not involve the functions of $p(s_t)$ and $p(s_{t+1}\mid s_t,a_t)$.

$$\begin{aligned}\nabla J(\theta) &= \int\nabla\pi(\tau;\theta)r(\tau)d\tau = \int\pi(\tau;\theta)\frac{\nabla\pi(\tau;\theta)}{\pi(\tau;\theta)}r(\tau)d\tau\\ &= \int\pi(\tau;\theta)\nabla\log\pi(\tau;\theta)r(\tau)d\tau = E_{\tau\sim\pi(\tau;\theta)}[\nabla\log\pi(\tau;\theta)r(\tau)]\\ &= E_{\tau\sim\pi(\tau;\theta)}[\nabla(\log p(s_t) + \sum_{t=1}^{T}\log\pi(a_t|\,s_t;\theta)\\ &\qquad + \sum_{t=1}^{T}\log p(s_{t+1}|\,s_t,a_t))r(\tau)]\\ &= E_{\tau\sim\pi(\tau;\theta)}[(\sum_{i=1}^{T}\nabla\log\pi(a_t|\,s_t;\theta)\sum_{t=1}^{T}\gamma^{t-1}r_t)]\end{aligned} \tag{9}$$

However, in the model-free environment, because the function of distribution is unknown, it is impossible to do the full expansion shown in (9). Only by performing actions in the environment, can we observe the state transition and obtain the reward. So inspired by the K-armed bandit [18], an alternative strategy is to sample multiple times and then calculate the average cumulative reward as an approximation of the expected one, which is called the Monte Carlo Method [13]. Equation (9) can be rewritten as (10):

$$\nabla J(\theta) \approx \frac{1}{N}\sum_{i=1}^{N}\left(\sum_{i=1}^{T}\nabla\log\pi(a_{i,t}\mid s_{i,t};\theta)\sum_{t=1}^{T}\gamma^{t-1}r_{i,t}\right) \tag{10}$$

where N is the sample times. So based on (10), the policy parameters θ are updated by performing the gradient-ascent algorithm step on $\nabla J(\theta)$ iteratively and gradually approximating the optimal policy $\pi^*(a_t|s_t;\theta)$. Equation (11) shows the update process of parameters θ.

$$\theta \leftarrow \theta + \alpha\sum_{k=1}^{t}\gamma^{k-1}r_k\nabla\log\pi(s_t,a_t;\theta) \tag{11}$$

Here α is a positive-definite step size. If the above formula can be achieved, then θ can usually be assured to converge to the optimal policy. Due to the space limitation, we omit the proof, more details can be found in [13].

The agent obtains revenue by interacting with the RTB bidding environment. First, the agent observes a state s_t from the environment and then based on the parametric policy $\pi(a_t|s_t;\theta)$, the agent chooses an action a_t upon the state s_t. After completing the bidding process, reward functions we design return reward r_t, and the agent observes the next state s_{t+1} from the environment. Finally, the agent obtains an episodic sequence from the environment, and then updates θ by performing a gradient ascent according to the cumulative discount reward $\sum_{k=1}^{t}\gamma^{k-1}r_k$ and corresponding gradient $\nabla\log\pi(a_t|s_t;\theta)$ for every state-action (s_t,a_t). Algorithm 1 shows the complete RLIB framework.

Algorithm 1 Reinforcement Learning to Intelligently Bid

Initialize episode numbers E
Initialize auction numbers T
Initialize learning rate α
Initialize weights θ of $\pi(a_t|s_t;\theta)$
for *episode* $=1$ to E **do**
  for $t=1$ to T **do**
    Observe state s_t
    If budget of $s_{t+1}\le 0$, then break
    Get action a_t from $\pi(a_t|s_t;\theta)$
    RL agent executes a_t in the environment
    observe state s_{t+1} and get reward r_t from *reward functions*
  end for
  Calculate $\sum_{t=1}^{T}\gamma^{t-1}r_t$ by current episode's sequence $\tau(s_1,a_1,r_1,s_2...s_T,a_T,r_T,s_{T+1})$
  for $t=1$ to T **do**
    $\theta \leftarrow \theta+\alpha\sum_{k=1}^{t}\gamma^{k-1}r_k\nabla\log\pi(a_t|s_t;\theta)$
  end for
end for

V. EVALUATION

A. Dataset

Our dataset is published by iPinYou[1], one of the leading DSP companies in the online advertising industry. For each impression, the bid logs contain the information of the user, advertiser, publisher, and the context. Also, the impression and click logs provide the info of bid price, paying the (market) price, user feedbacks (e.g., clicks), and whether the advertiser wins the auction. More details of the dataset can be found in [14]. In our experiments, we use the data of 2 days from 2013/06/06 and 2013/06/07 that the first day treated as the training set and the other treated as the testing set.

B. Evaluation Metrics and Experimental Setup

1) Evaluation metrics: The main objective of the agent is to optimize the KPI of ad campaigns under budget constraints (such as number of clicks, number of conversions or profits). In our experiment, we use the number of clicks we get as KPI. Also, we analyze other metrics, such as Cost-per-Click (CPC).

2) Evaluation process: The historical testing set contains a list of records. We consider that the arrival of impression is an episodic episode. Given the budget and bidding strategy, when a bidding request (a record) arrives, our bidding agent will generate a bid price in real-time according to the relevant features of the bidding request. If the bid price is higher or equal to the impression's market price, DSP will win the auction, then use the market price as the cost and get user clicks as the feedback, and finally update the remaining expected auction numbers and budget.

3) Budget constraints: if we set the budget to be the same as the original total cost in the testing logs, then just merely bidding as high as possible for each impression will exactly run out the budget and get all clicks [7]. In our work, we test KPI according to various budget constraints and run evaluation tests separately. Finally, we use 1/2, 1/4, 1/8, 1/16 of the original total cost in the test logs as the budget.

4) Hyper-parameter setting: we provide some key hyper-parameter settings to facilitate the implementation of our method, as shown in table I. In RLIB, we adopt a fully connected neural network as the action selection policy $\pi(a_t|s_t;\theta)$, which contains 100 nodes for the hidden layer and the learning rate used by the gradient ascent algorithm is set to 0.0001. It is worth noting that for all the baseline bidding strategies, the hyperparameter settings and network structure provided by the corresponding papers are employed.

TABLE I. HYPER-PARAMETER SETTING

Hyper-parameter	Description
0.0001	The learning rate used by gradient ascent
300(cent)	Cost per click
Tanh	The activation function of hidden layers
100	Number of neurons in fully connective layer

[1] iPinYou Dataset website: http://data.computational-advertising.org/

C. Baseline methods

In the experiment, we will compare the baselines with our proposed strategy. During the experiment, the training data is used to adjust the parameters of each bidding strategy (all bidding strategies use the same prediction model (FM) to predict the click-through rate for each impression). The baseline methods in our experiments include:

1) Deep Reinforcement learning to bid(DRLB): a budget-constrained bidding strategy by leveraging value-based model-free reinforcement learning [8].

2) Reinforcement learning to Intelligently bid(RLIB): a policy-based model-free reinforcement learning bidding strategy under budget constraints, which we proposed with four reward functions.

3) Reinforcement learning to bid with immediate reward (RLBIR): to reflect the performance of the four reward functions we designed, we conduct a comparative experiment that set r_t in Algorithm 1 as the immediate reward (i.e., click).

D. Offline Evaluation

In this subsection, we conduct experiments on a public real-world dataset iPinYou and evaluate the performance of our proposed *Reinforcement Learning to Intelligently Bid* framework by comparing with two state-of-the-art baselines. Also, to analyze the performance comparison under different budget constraints, we set the budget as 1/16, 1/8, 1/4, and 1/2 of the original total cost in history, respectively.

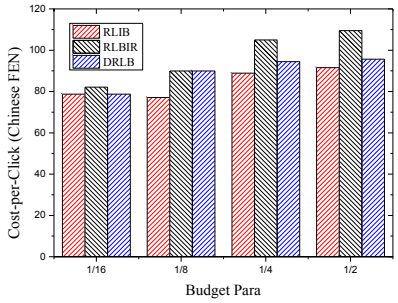

Fig. 3. Performance comparation with CPC

Fig. 3 shows the CPC values of the three bidding strategies under different budget constraints. We observe that the CPC of each bidding strategy increases as the budget increases. This is because when the budget is low, the optimal bidding strategy will be more inclined to push the budget on the impressions with the low market prices; when the budget is higher, the budget will be allocated on more impressions with high quality but relatively high market prices, which results in the higher each campaign's CPC. Among all bidding strategies in the experiments, our method is able to obtain the lowest CPC, followed by RLBIR, and the highest is DRLB. And the results show that the policy-based solution is better than the traditional solution based on the value function in RTB.

Fig. 4 shows the total clicks for each bidding strategy under different budget constraints. We can observe that our method gets the highest number of clicks, and while we capture fewer clicks as the budget decreases, it is consistently superior to the other two baseline strategies. The performance comparison of total clicks and CPC under different budget constraints are reported in table II.

TABLE II. PERFORMANCE COMPARISON AMONG DIFFERENT BIDDING STRATIGES (BUDGET = 30228554)

	RLIB				DRLB				RLBIR			
Para	**Bids**	**Imps**	**Clicks**	**CPC(cent)**	**Bids**	**Imps**	**Clicks**	**CPC(cent)**	**Bids**	**Imps**	**Clicks**	**CPC(cent)**
1/2	317668	293024	165	91.60	235235	234850	138	109.52	353565	288045	158	95.66
1/4	154914	142747	85	88.91	119453	115977	72	104.96	169199	141375	80	94.46
1/8	76963	71684	49	77.11	60394	59884	42	89.97	83995	71031	42	89.97
1/16	39160	36633	24	78.72	31687	31248	23	82.14	42431	36278	24	78.72

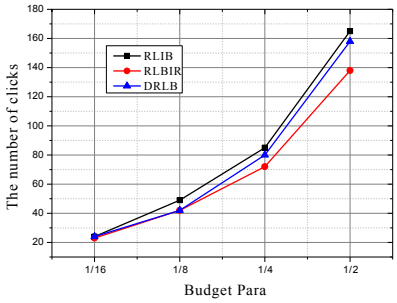

Fig. 4. Performance comparation with clicks

VI. CONCLUSION

As the most critical component in DSP, the bidding strategy has always been a hot topic in the field of online advertising. In this paper, we propose an intelligent bidding strategy based on a policy-based model-free RL framework, to help ad campaign determine the bid prices of all available impressions. First, we formulate the bidding problem as a Markov Decision Process and then learn the optimal bidding policy for each campaign by using the policy gradient algorithm. There are two characteristics of our framework: one is that policy gradient algorithm is introduced to learn the bidding strategy, which can overcome the convergence problem in other reinforcement bidding learning with value-based methods; the other is that we design four reward functions for different auction results and user feedback, taking into account the cost and benefit. Finally, we evaluate the performance of the proposed bidding strategy based on a real-world dataset, and the experimental results have demonstrated the superior performance and high efficiency compared to the state-of-the-art method. In the future work, we will try to deal with the problem that using reinforcement learning is hard to directly derive the proper impression-level bid price. And due to the highly dynamic environment of the auction market, we will try to incorporate budget management mechanism into the reinforcement bidding learning framework.

ACKNOWLEDGMENT

This work was supported in part by the National Natural Science Foundation of China under Grant 61202445 and Grant 61472064, the Fundamental Research Funds for the Central Universities under Grant ZYGX2016J096.

REFERENCES

[1] Yuan S, Wang J, Zhao X. "Real-time bidding for online advertising: measurement and analysis," in Proceedings of the Seventh International Workshop on Data Mining for Online Advertising. ACM, Chicago, Illinois, U.S.A, 2013.

[2] Wang J, Yuan S. "Real-time bidding: A new frontier of computational advertising research," in Proceedings of the Eighth ACM International Conference on Web Search and Data Mining. ACM, 2015, 415–416.

[3] J Wang, W. Zhang, and S Yuan, "Display advertising with real-time bidding (RTB) and behavioural targeting," Foundations and Trends in Information Retrieval, 2017, vol. 11.no. 4-5: pp. 297 - 435.

[4] V. Krishna. Auction theory. Academic press, 2009.

[5] H Cai, K Ren, and W Zhang, "Real-Time bidding by reinforcement learning in display advertising," in WSDM '17, New York, NY, USA : ACM, 2017: 661-670.

[6] Perlich C, Dalessandro B, and Hook R, "Bid optimizing and inventory scoring in targeted online advertising," in KDD '12, New York, NY, USA : ACM, 2012: 804-812.

[7] W Zhang, S Yuan, and J Wang, "Optimal Real-time bidding for display advertising," in KDD '14, New York, NY, USA : ACM, 2014: 1077-1086.

[8] D Wu, X Chen, X Yang, H Wang, Q Tan, and X Zhang, "Budget constrained bidding by Model-free reinforcement learning in display advertising," in CIKM '18, New York, NY, USA: ACM, 2018: 1443-1451.

[9] Lillicrap, T. P, Hunt J. J, Pritzel A, Heess N, Erez T, and Tassa Y, "Continuous control with deep reinforcement learning," Computer Science 8.6(2015): A187.

[10] Gordon G J, "Stable function approximation in dynamic programming," Carnegie Mellon University, 1995:261-268.

[11] Sutton R S, McAllester D, and Singh S, "Policy gradient methods for reinforcement learning with function approximation," in Advances in Neural Information Processing Systems, Denver, CO, United states, 2000: 1057 -1 106.

[12] Ta A, "Factorization machines with follow-the-regularized-leader for CTR prediction in display advertising," in Proceedings - 2015 IEEE International Conference on Big Data, IEEE Big Data 2015, Santa Clara, CA, United states, 2015: 2 289-2889.

[13] Sutton R S, Reinforcement learning: An introduction. IEEE Transactions on Neural Networks, 1998.

[14] H Liao, L Peng, and Z Liu, "iPinyou global RTB bidding algorithm competition dataset," in Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, NY, United states, 2014: Microsoft.

[15] C. Watkins, P. Dayan, "Technical note Q-Learning," Machine Learning, vol. 8, pp. 279-292, 1992.

[16] Mnih V , Kavukcuoglu K , Silver D, Graves A, Antonoglou I, Wierstra D and Riedmiller M. A., "Playing Atari with Deep Reinforcement Learning," Computer Science, 2013.

[17] Rummery, G. A, "On-line Q-learning Using Connectionist Systems," Technical Report, 1994.

[18] Mohri, Mehryar, "Multi-armed bandit algorithms and empirical evaluation," in Proceedings of the 16th European Conference on Machine Learning (ECML), pp. 437-448, Porto, Portugal.

[19] Paul Grigas, Alfonso Lobos, Zheng Wen, and Kuang-chih Lee, "Profit Maximization for Online Advertising Demand-Side Platforms," in Proceedings of the ADKDD'17, 2017.