Real-Time Bidding with Soft Actor-Critic Reinforcement Learning in Display Advertising

Daria Yakovleva, Artem Popov, Andrey Filchenkov
ITMO University
St. Petersburg, Russia
{dariyakovleva, artpop92}@gmail.com, aaafil@mail.ru

***Abstract*—The main task of advertising companies is to sell goods and services interesting to the user. Online auctions are the main mechanism for selecting ads to the user. Dynamic bidding allows advertiser to automatically calculate the bid that is profitable to set to maximize goals (for example, the number of clicks on an ad), depending on the user who sees the ad. In this case the advertiser must specify the budget of the ad and the optimization goal. During the advertising campaign the bid for each impression will be calculated by a special algorithm. In this paper, we propose a novel algorithm for calculating the dynamic bid for each impression of the ad in order to maximize the advertiser's goals, which takes into account settings of the advertising campaign, budget, the ad lifetime and other parameters. This task is formulated as reinforcement learning problem, where states are the status of auction and parameters of the advertising campaign, the actions are bidding for each ad based on the input state. Every ad has an agent who observes the states all the time and calculates the bid for the impression. We evaluated the proposed model on real advertising campaigns in a large social network. Our method achieved average 26% improvement in comparison with the state-of-the-art approach.**

I. Introduction

Online display advertising ecosystems play a key role in connecting publishers and marketers [1], resulting in the constant growth of the market share they seize [2], [3]. An important challenge such ecosystems meet is the development of a mechanism for matching the most relevant users with the most profitable advertisement.

The majority of online display ads are served through real- time bidding (RTB), where each ad display impression is auctioned off in real-time when it is just being generated from a user visit. To place an ad automatically and optimally, it is critical for advertisers to devise a learning algorithm to cleverly bid an ad impression in real-time [4].

RTB mechanism has the most significant progress in recent years in online display advertising for buying and selling ads[5]. Many relevant advertising campaigns are selected to the user per each impression. The campaign with the highest bid wins the auction. Typically, the price of the target action, click or conversion, is static. It is set by an advertiser and does not depend on a user to whom the advertisement will be shown. Being simple, it shows very low flexibility and adaptivity to context and market conditions.

These limitations are overcome with the concept of dynamic pricing assuming that the cost of an action depends on the user and the goals of the advertiser. In this case, the cost is automatically calculated to maximize the advertiser's goals such as the number of clicks on the ad. Optimizing cost-per-impression (oCPM) [6] allows advertisers to show ads for people who most likely makes the target action.

Using an automatic bid the advertiser must set the budget of the ad and optimization goal (click the link, watch the video, purchase, etc.), the bid for each impression during the advertising campaign will be calculated by a special algorithm. The advertiser can also set the targeting audience settings include age, place of residence, users interests, activity in communities.

In this paper, we suggest an algorithm for budget-constrained dynamic bidding, which increases the effectiveness of advertising campaign in online display advertising and takes into account the specific parameters of the advertising campaign, the budget of the ad and the goals of the advertiser.

We formulate the budget-constrained bidding as the reinforcement learning problem, where states are the status of auction and parameters of the advertising campaign, the actions are bidding for each ad based on the input state. Every ad has an agent who observes the states all the time and calculates the bid for the impression.

The rest of the paper is organized as follows. In Section II, we overview related papers. In Section III, we propose a novel algorithm for budget-constrainted dynamic bidding. Section IV contains results of empirical research of the suggested algorithm performance and its comparison with baseline algorithms. In Section V, we analyze and discuss the results of the study and suggest several ways to improve the algorithm. A conclusion is in a Section VI.

II. Related Work

In this Section, we briefly review papers on the RTB system including some basic and state-of-the-art bidding strategies.

A. Basic algorithm

One of the most common strategies is linear bidding $LinBid$ [7], where the bid value for each impression is linearly dependent on the predicted click probability of the user per the ad.

In online bidding, there are strict time limits, where the bid calculation has to take less than 100 milliseconds after receiving a request. Also, the auction and ads competition per impression change so fast and the advertiser must adapt these changes.

Linear bidding uses Click-through rate (CTR) as θ, which means the probability of the user click per ad. Calculation of k-th bid b_k depends on the historical CTR, the current CTR estimation ϕ_k and the custom parameter b_0:

$$b_k = b_0 \cdot \frac{\phi_k}{\phi_{average}}.$$

B. Analytic solution

One of the most significant articles in the field of optimal bid calculation was published by Weinan Zhang et al. [8]. The authors analytically show that the optimal bid function has a non-linear relationship with the impression level evaluation such as the CTR and the conversion rate. This paper shows that an optimal bidding strategy should try to bid more impressions rather than focus on a small set of high valued (leading to conversion) impressions.

Also, the authors formulated functional optimization problem with constraints. The goal is to find a bidding function that maximizes the total gain from impressions with the budget constraint.

Let

- x be a bid request represented by its features;
- $p_x(x)$ be the probability density function of x;
- $\theta(x)$ be a predicted Key Performance Indicators (KPI) if winning the auction of x. It could be the CTR or conversion rate (CVR);
- $b(\theta(x))$ be a bidding strategy function;
- B be the campaign budget;
- N_T be an estimated number of bid requests during the lifetime T of the budget;
- $w(b(\theta(x)))$ be the probability of winning the bid request x with bid price $b(\theta(x))$.

The authors rely on two assumptions to solve the optimisation problem. First, they assume that $b(\theta(x))$ does not depend on x directly, but depends throw θ. Second, $w(b)$ doesn't depend on x directly, but depends throw b.

Mathematically, optimal bid generation problem is formulated as a functional optimisation problem:

$$b(\theta)_{ORTB} = \arg\max_{b()} N_T \int_x \theta(x) w(b(\theta(x))) p_x(x) dx$$
$$\text{subject to } N_T \int_x b(\theta(x)) w(b(\theta(x))) p_x(x) dx \leq B$$

This allows to obtain an equation for the optimal bidding function b_{ORTB1} of two parameters c and λ, which we will use later:

$$b_{ORTB1}(\theta) = \sqrt{\frac{c}{\lambda}\theta + c^2} - c. \quad (1)$$

As shown in Fig. II-B, compared with $LinBid$, b_{ORTB1} bids higher when the estimated KPI is low, which means optimal real-time bidding (ORTB) allocates more budget on the cases with low bid price and thus low cost.

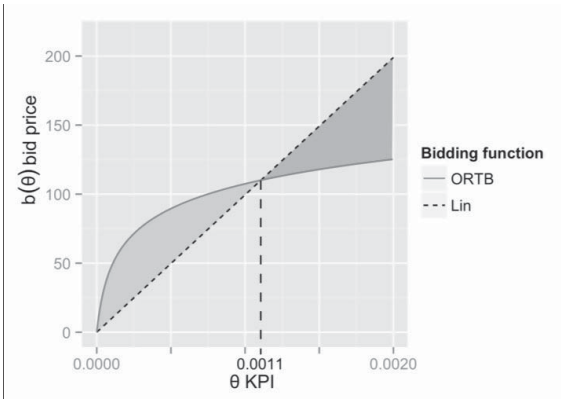

Fig. 1. ORTB1 and linear bidding comparison [8]

C. Reinforcement learning approaches

In paper [4], decision making process is formulated as a reinforcement learning problem, where states contain auction information and advertising campaign parameters in real time. At each moment the ad agent observes states that contain the current campaign parameters such as the remaining lifetime and budget and the current bid request for a specific impression. The agents action is to calculate the best bid for an impression.

The authors build a Markov Decision Process (MDP) [9] framework for learning the optimal bidding policy to optimize advertising performance. The value of each state is calculated by performing dynamic programming.

RTB strategy is described below. The advertiser receives an online bid request. The first step to take is to estimate utility, that is the probability that the user will perform the target action provided that the ad wins the auction.

The market price of other advertisers is predicted using the bid landscape forecasting method to evaluate the probability of winning an auction with a given bid. Having an estimate of the users benefit and price distribution, the agent calculates the impression bid based on information about the remaining campaign budget. After participation in the auction, the agent receives a reward defined by if it reached the target action specified in the parameters of the advertisement.

A crucial drawback of this model-based RL approach (RLB) [10] is the requirement of storing the state transition matrix and using dynamic programming algorithms, whose computational cost is unacceptable in real-world advertising platforms.

The state-of-the-art approach described in paper [11] is trying to overcome this drawback by approaching the problem using a combination of reinforcement learning and deep learning. The authors adopt the widely used model-free approach in multi-agent reinforcement learning algorithm. Agent action is sequential regulation parameter λ instead of directly bid generation.

One of the solutions uses $\lambda = f(\text{budget})$, because the speed of budget spending depends on λ, therefore the speed of budget spending can be used for λ regularization. Bidding function is

thus as follows:

$$b_t, k = \frac{\phi_{t,k}}{\lambda_t}.$$

The agent is trying to learn and adapt to a highly unstable environment in order to make λ always close to the optimal value.

To approximate the Q-function, the authors use deep Q-network (DQN). This approach avoids the computational complexity. This is why we use this approach as the baseline for results comparison.

D. Other approaches for problem solution

The article [12] proposed a similar solution to the one presented in [4]. However, their system additionally has guarantee contracts between the platform and advertising agencies, according to which the platform is required to show agency ads at least for a certain budget. This slightly complicates the task, but in general, the solution is similar.

The article [13] presents a distributed learning algorithm with reinforcement learning for solving the dynamic bidding problem. The proposed algorithm was tested on the Taobao advertising platform. Their solution has specifics for market platforms, the purpose of which is the sale of goods. Therefore, they pay a lot of attention to the algorithm for recommending products to users. Also, they have a larger number of sellers in comparison with RTB systems, because any user can become a seller of goods, so in order not to create an agent for each seller, they solve the problem of sellers clustering and creating an agent for each of the clusters.

The article [14] proposed an iterative method for calculating bids in order to maximize the total number of clicks on an advertising campaign. First, the authors tried to optimally divide the budget between different advertising formats depending on the quality of each format KPI. After dividing the budget, they used a model for predicting the probability of winning an auction with a given bid. Then they used the gradient descent method to find the best rates.

The article [15] also presented a model for calculating the optimal bid, which used the latest data from winning auctions and bids, as well as three components: the probability of a positive user reaction, the prediction of user reach at a given bid, and the bid calculation strategy as a single objective function. Gradient descent was used to optimize these things.

A general scheme of the approach proposed in [15] is presented in Fig. II-D.

The article [16] presented an algorithm for choosing the most profitable Supply-Side Platform (SSP) in order to select an ad for display from it. The algorithm is based on Thompson sampling [17].

III. Proposed Algorithm

In this Section, we describe the proposed algorithm as well as some other attempts we tried.

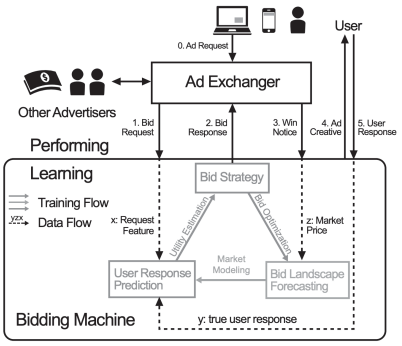

Fig. 2. Bidding machine framework [15]

A. Problem Statement

This work focuses on creating an impression bid strategy for an advertising campaign to maximize goals, such as clicks or conversions.

Consider an advertising campaign with budget B and the goal of maximizing is the number of clicks. The lifetime of the advertising campaign is divided into T episodes, each episode includes K ad auctions for different users.

Assume that there is a separate model predicting the click probability of a specific user on the advertisement $\theta_{ad,user} = \theta_{t,k}$.

The task is to find a bid function for impressions $b(ad, user, \theta)$ in order to maximize the number of clicks under the advertiser budget constraint B. In terms of reinforced learning, the challenge is to develop an agent that will automatically determine the best bid for an impression that is beneficial for the advertiser to maximize a given goal.

B. Online Bidding algorithm

The dynamic bid calculation algorithm is based on a reinforcement learning approach. Each state S_t is calculated after each episode. It is described by the current parameters of the advertising campaign (for example, the remaining budget) and indicators of the campaigns performance in the previous episode (for example, the number of win impressions). The agents action is to increase or decrease the λ_t parameter, which determines the final bid for an impression in the next episode. Agent reward is the CTR of win impressions.

The auction state space S contains all kinds of tuples S_t.

The state S_t is described by the following variables:

- episode t;
- remaining budget B_t;
- the remaining number of episodes ROL_t, which can be considered as the remaining number of possibilities to change the parameter λ;
- budget spend rate $\beta_t = \dfrac{B_{t-1} - B_t}{B_{t-1}}$;
- percentage of winning auctions WR_t;
- average CPM of winning impressions CPM_t in the previous episode t;

- total winning clicks r_t in the previous episode t.

Thus, the state is described as

$$S_t = (t, B_t, ROL_t, \beta_t, CPM_t, WR_t, r_t). \quad (2)$$

The agent considers the relevant campaign parameters and decides whether to increase or decrease the λ_t parameter.

For example, if the budget is almost spent and there is still the possibility of changing the budget, then the agent should increase λ and thus generate bids less aggressively.

Action space is possible corrections of λ_t: $A = \{8\%, 3\%, 1\%, 0, 1\%, 3\%, 8\%\}$:

$$\lambda_t = \lambda_{t-1}(1 + A_t), A_t \in A. \quad (3)$$

The reward R_{t+1} after performing the action A_t in the state S_t is defined as

$$R_{t+1} = \sum_{k=1}^{K} X_{t,k}\phi_{t,k}, \quad (4)$$

where $X_{t,k} = 1$ if k-th impression was won and $X_{t,k} = 0$ otherwise. ϕ_t, k is CTR estimation for k-th bid.

The interaction of the agent and the environment is shown in the Fig. III-B.

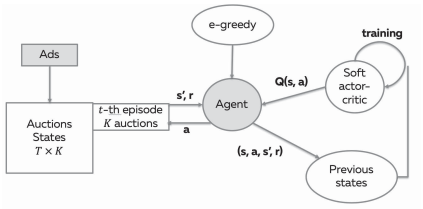

Fig. 3. Algorithm workflow

The pseudo-code for participation in auctions during one episode is presented below.

Listing for ad participation in auctions

```
AUCTIONS EPISODE (B_t, CTR, ROL)
01    Get λ from agent
02    bud_left = B_t
03    CPM = 0
04    WR = 0
05    reward = 0
06    For (i = 1...K)
07      bid[i] = calculateBid(ctr[i], λ)
08      If (bid[i] > winbid[i] and bid[i] > budget_left)
09        budget_left = budget_left − winbid[i]
10        reward = reward + ctr[i]
11        WR = WR + 1/K
12        CPM = CPM + winbid[i]/K
13      EndIf
14    EndFor
15  budget_spent = budget_left/budget
16  β[t] = (budget − budget_left)/budget
17  State = (budget_spent, ROL, β[t], CPM, WR, reward)
```

C. Dynamic bid calculation

As we described in details in Section II, the analytical solution was suggested for impression bid evaluation in the paper [8]. We will reuse formula in Eq. 1 in our algorithm.

Let $b_{t,k}$ be the bid at the step k at time t and c is custom parameter. In our algorithm, it is determined by the following formula:

$$b_{t,k}(\theta) = \sqrt{\frac{c}{\lambda_t}\theta_{t,k} + c^2} - c$$

Using this formula helps the agent instead of overpaying for expensive impressions, buying less expensive impressions with a comparable CTR, thus spending the budget more optimally.

D. Soft-actor critic approach for agent learning

For each state S_t, the agent chooses an action using Q-function evaluating the reward.

For agent training, a soft actor-critic approach is used, which was first described in [18].

The purpose of this approach is not only to maximize rewards, but also entropy when choosing actions. Therefore, the agent is able to make proper bid in more random situations. This is useful because the auction is very dynamic, for example, during the holidays, competition in the auction increases significantly.

Another advantage of this approach is that it takes less samples for training and it is less sensitive for hyper-parameters tuning.

The agent learning algorithm is described below:

- Get mini-batch from replay memory
- Get reward y_i

$$\begin{cases} y_i = r_i, & s_i \text{ terminated} \\ r_i + \gamma max_{a'} V(s_{j+1}) & \text{otherwise} \end{cases} \quad (5)$$

- Update the Q-function: SGD of $(y_i - Q(s_i, a_j; \theta))^2$
- Build $y_s = Q(s, a) - \alpha\pi_\theta(a|s)$, a — $\pi_\theta(\dot{|}s)$
- Update V-function: SGD of $(y_s - V(s))^2$
- Update $policy$-function: $Q(s, a) - \alpha \log \pi_\theta(a|s)$

This approach has improved the quality of the agents performance, the comparison and results are presented in the chapter IV.

E. Using user similarity in the bid calculation

Note that the model for predicting the probability of ad click is general and used for all ads. It may not take into account some local specific features of the ad or its target audience, for example, due to large training dataset.

Accordingly, we would like to have the opportunity for a particular ad to identify common significant features of users who are already interested in this ad. It can be obtained using a similar user search algorithm (lookalike), which implicitly

finds common features of a positive samples and orders other users according to similarity to positive samples.

Consider users from the training dataset who have already clicked on the ad. We will evaluate the similarity of other users from the target audience of the ad to this set using the $lookalike$ algorithm. Thus, for each target we get the number l, which reflects its similarity to users who have already clicked on the ad.

We normalize the vector of similarity numbers of the ad target audience using z-score:

$$l_i = \frac{l_i - \text{SD}(L)}{\text{mean}(L)}, \forall i,$$

where SD is standard deviation.

After normalizing score for each user, we obtain estimation of how much the user is more similar than average user to positive samples. Due to we are only interested in similar users, we will get rid of negative similarity scores:

$$l_i = \max(l_i, 0).$$

We will use this estimation as a correction coefficient for the predicted click probability θ in the bid formula:

$$b_{t,k}(\theta) = \sqrt{\frac{c}{\lambda_t}\theta_{t,k} \cdot l + c^2} - c.$$

Also, for each episode, we calculate how much the users whose auctions we won are similar to a positive samples. Add this information as an additional state parameter.

$$S_t = (t, B_t, ROL_t, \beta_t, CPM_t, WR_t, r_t, Lal_t),$$

where $Lal = \text{mean}(l_i), i \in Impressions.$

Thus, using the search algorithm for similar users, we can expand features set that describes the user's interest to the ad, using information about the similarity of users.

F. Alternative solutions

In the process of searching for a new solution, alternative methods of calculating the dynamic bids and training agent were considered.

1. b_{ortb2}, the second analytic solution, described in the paper [8]:

$$b_{ortb2}(\theta) = c \cdot \left(\left(\frac{\theta + \sqrt{c^2\lambda^2 + \theta^2}}{c\lambda} \right)^{\frac{1}{3}} - \left(\frac{c\lambda}{\theta + \sqrt{c^2\lambda^2 + \theta^2}} \right)^{\frac{1}{3}} \right)$$

This solution is proposed to be used for auctions with high competition. Comparison showed that this formula does not improve the main indicators of the quality of the algorithm.

2. Deep Q-network + $b_{ortb1}(\theta)$: We tried to use DQN instead of Soft Actor-Critic for agent training.

3. Actions space $A = \{1, 4; 1, 6; 1, 8; 2; 2, 2; 2, 4; 2, 6\}$: We tried to use an alternative action space. Instead of each action being an increase or decrease of λ, let the action be the optimal multiplier for λ_0, which is most profitable to calculate the bet in the next episode: $\lambda = \lambda_0 \cdot a_{best}$

4. Change the bid proportionally to similarity score: We tried to apply the information about the user's similarity not to the CTR, but directly on the bid for the impression. The result of the algorithm has become worse, probably because the bid in this case depends on the similarity of the user, which is not always the main indicator for increasing the cost per impression.

$$b_{t,k}(\theta) = (\sqrt{\frac{c}{\lambda_t}\theta_{t,k} + c^2} - c) \cdot Lal.$$

5. Another normalization of the user similarity: The proposed solution uses z-score for normalization. We also tried to use another method: $l = \frac{l - \min L}{\max L - \min L}$.

6. Adding the parameter $L_{percentile}$ = 70% percentile of user's similarity among the selected impressions of the last episode: Instead of L_{mean}, we tried to use $L_{percentile}$ because of assumption that it was more important not how much the users whose impressions were won are on average, but how much the chosen bidding strategy helped to win auction of most similar users.

We found these methods less efficient than main solution for most of ad campaigns. All the alternative solutions did not give a noticeable improvement in the main indicator, the number of win clicks. Due to the paper volume limitation and unimportance of the failures given that we have an efficient algorithm, we decided not to include these results as a Table, only just by mentioning the directions of attempts.

G. The scheme of the proposed algorithm

The final algorithm using the described approaches is presented below.

SAC Algorithm

```
SAC()
01    Initialize replay memory D with size
n_d
02    Initialize Q_local random values
03    Initialize Q_target random values
04    For (episode = 1...N)
05      Initialize λ_0
06      For (k = 1...K)
07        Set the bid b_{0,k}(θ) = sqrt((c/λ_0)θ_{0,k} · l + c^2) − c
08      EndFor
09      For (k = 1...T)
10        We are in state s_t
12        Choose optimal action a_t
13        Set λ_t = λ_{t−1}(1 + a_t)
14        For (k = 1...K)
15          Set bid b_{t,k}(θ) = sqrt((c/λ_t)θ_{t,k} · l + c^2) − c
16          Use the bid b_{t,k} at auctions
17        EndFor
18        Get reward r_t for the episode
19        Move to the next state s_{t+1}
20        Save the tuple (s_t, a_t, r_t, s_{t+1}) to
replay memory D
21        Get mini-batch from replay memory
D and update soft-actor-critic
23      EndFor
```

```
24    EndFor
```

The algorithm uses the formula from Eq. 1 for calculating bid, the agent is trained based on the soft actor-critic approach, and an algorithm for searching similar users is used to expand the user feature space.

IV. Experiments and Results

In this Section, we evaluate the performance of the proposed SAC algorithm and compare it with baseline algorithms. As algorithms for comparison, we use the linear random bid ($LinBid$) algorithm, which is described in Section II and the algorithm from paper [11] (DRLB)n.

At the beginning of the Section, the results of comparing each improvement of the algorithm is presented. After that the proposed solution is compared with the current published results. In addition, a comparison of the proposed algorithm with alternative approaches, also proposed by the authors, is presented. At the end of the Section we conduct analysis of the agents performance on new advertising campaigns whose data are not presented in the training set.

A. Metrics for comparison

The main metrics that will be used for comparison are the ratio of the number of won impressions to all impressions, the percent of the spent budget, effective cost-per-click (effective CPC eCPC), and effective cost-per-impression (eCPI):

- **impressions** is the number of won impressions
- **budget** is spent budget
- **clicks** is the number of won clicks
- **win rate** is the ratio of won impressions to the number of auctions
- **CTR** is the ratio of won clicks to won views
- **eCPC** is effective cost per click
- **eCPI** is effective cost per view
- number of win clicks with limited budget $b_0 \in \{1/64; 1/32; 1/16; 1/8; 1/4; 1/2; 1\}$

B. Datasets

Two datasets are used to test the algorithms: an open dataset iPinYou and a dataset provided by the social network VK.

1) iPinYou dataset: The main dataset for comparison is the dataset from the company iPinYou [19]. The dataset makes possible to compare the results with other articles.

The dataset contains 9 different ads, with several days duration each. Events for each dataset are divided into training (train) and testing (test) subsets. In total the dataset for training contains about 15 million impressions and the dataset for testing contains about 4 million impressions. The dataset is described in Table I.

Since the agent calculates the impression bid using the CTR estimation for the impression and after that selects the action taking into account the remaining budget, the agent needs bids with which impressions were won, CTR estimates for each impression, budget size and the number of impressions for each advertisement.

TABLE I. iPinYou dataset

Ad number	Imprs in train	Clicks in train	Imprs in test	Clicks in test
1458	3083056	2454	614638	543
2259	835556	280	417197	131
2261	687617	207	343862	97
2821	1322561	843	661964	394
2997	312437	1386	150063	533
3358	1742104	1358	300928	339
3386	2847802	2076	545421	496
3427	2593765	1926	536795	395
3476	1970360	1027	523848	302

2) Dataset from VK: The dataset contains anonymized ad impression data. The ads are different: high and low CPC, high and low CTR, wide and narrow audience. A Number of impressions in the dataset is about 7.5 million.

The dataset is described in Table II and Table III.

TABLE II. Dataset from VK, part 1

Ad number	Imprs in train	Clicks in train	Imprs in test	Clicks in test
1	394314	3862	168992	1397
2	505812	1371	216778	797
3	594506	556	254789	264
4	1358161	11245	582070	3371
5	1071068	6635	459030	1939
6	735537	779	315231	221
7	75556	1100	32382	420
8	54145	413	23205	162
9	56850	171	24365	63
10	142450	3739	61051	1347
11	77498	997	33214	245
12	86319	380	36995	158

TABLE III. Dataset from VK, part 2

Ad number	CTR	CPC, rubles
1	0,9%	15
2	0,3%	50
3	0,1%	60,81
4	0,8%	12,23
5	0,5%	3,3
6	0,09%	6,62
7	1,4%	20
8	0,7%	65,2
9	0,3%	64
10	2,4%	1
11	1,1%	4,3
12	0,4%	2,1

The most difficult task is to successfully train the agent. Many different parameters affect the learning outcome.

The article [11] does not explain in detail how they initialized the budget during the training. The iPinYou dataset contains the budget for each ad campaign. It is also not clear how to split and initialize the budget for each episode in one campaign. One of the goals of the agent is the optimal management and spending of the budget. Therefore, it is important how to set and distribute the budget.

The article [20] uses the specified budget for training B_{train} in the iPinYou dataset, after that they scale this budget for training in proportion to the ratio of impressions in the test set N_{test} to the number of impressions in the training set N_{train}. They also use parameter b_0 to analyze the agent

behavior with different budgets. Thus, the test budget is calculated by the formula:

$$B_{test} = b_0 \cdot \frac{N_{test}}{N_{train}} \cdot B_{train} \quad (6)$$

C. Algorithm parameters

The main problem of initialization in reinforcement learning is that the incorrect initialization of some parameters can lead to inefficient learning, in which the agent quickly converges into a suboptimal solution or simply cannot learn anything.

For example, if λ_0 parameter is too small, the agent can set the bids so aggressively that it wins all auctions and quickly spends a given budget. In this case, the study of space does not give an effect, and the agent does not learn how to regulate the budget so much as not to spend it in an instant.

On the other hand, if the parameter λ_0 is too large, then the agent will fall into situations where you need to spend a significant part of the budget in order to get a noticeable reward. This is closely related to the speed of learning. If an agent wins several auctions per episode, it will be remembered.

After that, the agent will use this example for training with other examples in which more auctions have been won, but with lower bids and lower rewards. In other words, even if an agent is trading more aggressively, this experience may have a negligible impact on training, especially at a low training speed.

Custom parameters for agent training and testing are described below. For some of them, we set the values that are described in the articles for comparison, others we will select.

- the number of auctions per episode K
- training speed α
- start λ_0 in the bidding formula
- scaling budget parameter b_0
- frequency of training target network $C = 100$
- replay memory size $n_D = 10^5$
- the number of episodes $T = 100$
- discount factor $\gamma = 1$

All tests with iPinYou dataset are performed with scalable budget parameter $b_0 = 1/32$.

All tests with the social network dataset are performed with a scalable budget parameter $b_0 = 1/16$.

D. Results of each improvement

Three main ideas were proposed in this paper:

- Using optimal bidding formula b_{ortb1} ($ortb1$)
- Using Soft Actor-Critic approach for agent training (sac)
- Using information about users similarity in bidding strategy and learning agent (lal)

As a basic solution, we will use agent training using Deep Q-network (DQN) and bid calculation using the formula $bid = \frac{CTR}{\lambda}$.

Table IV and Table V show how each of the ideas improves the quality of the algorithm for each dataset. The iPinYou dataset does not contain information about user features, so their similarity cannot be calculated.

From Table IV it can be seen that the $ortb1$ approach works better in 55% of cases relative to the basic solution, the sac approach works better in 55% of cases, while for other campaigns, both approaches improve the agents result in 89% of cases.

Table V shows that the $ortb1$ approach works better in 50% of cases relative to the basic solution, the sac approach works better in 83% of cases, the lal approach in 50% of advertising campaigns, in total solutions improve the agents performance in 100% of cases. It can be seen that each of the approaches contributes to the efficiency of the agent.

TABLE IV. IPINYOU. IMPROVEMENTS COMPARISON. CLICKS AND ECPC

Ad id	no improvements	ortb1	sac	ortb1 + sac
1458	458 (1,46)	417 (0,24)	**463** (2,47)	461 (2,86)
2259	9 (104,9)	13 (77,69)	9 (116,23)	**15** (67,33)
2261	**15** (56,23)	8 (104,06)	13 (64,87)	9 (92,50)
2821	30 (55,31)	26 (32,58)	**40** (33,77)	**40** (34,18)
2997	66 (3,77)	**88** (3,96)	63 (6,01)	**88** (3,76)
3358	190 (2,09)	197 (1,89)	204 (2,77)	**212** (3,33)
3386	43 (2,93)	58 (22,54)	**90** (14,10)	89 (14,69)
3427	290 (2,83)	262 (0,63)	283 (2,12)	**294** (4,16)
3476	133 (3,26)	157 (7,54)	164 (4,02)	**170** (4,72)

The results of the agent are presented in Fig. IV-D-IV-D. They show the clicks earned by the agent for the epochs and the total reward of the agent (CTR).

The gentle part of the graph in recent episodes is due to the spending of the entire budget by the agent. It can be seen that the agent $ortb1 + sac + lal$ spends the budget more optimally, does not spend it immediately, and this helps him get more rewards, win more clicks.

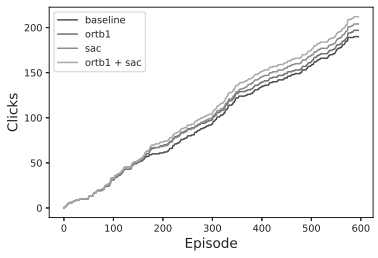

Fig. 4. Ad 3358. Win click

E. Comparison with actual solutions

Table VI and Table VII show the final results on the iPinYou open dataset.

According to testing results on the iPinYou dataset, the new SAC algorithm in some campaigns wins more clicks (30% of campaigns), in some it receives a lower cost per click (55%

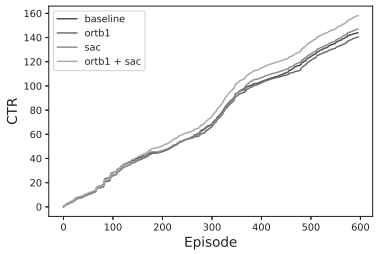

Fig. 5. Ad 3358. Agent reward

TABLE V. VK DATASET. IMPROVEMENTS COMPARISON. CLICKS AND ECPC

Ad id	baseline	ortb1	sac	lal	ortb1+sac+lal
1	68 (11,22)	55 (9,04)	69 (11,06)	51 (13,96)	**90** (8,48)
2	57 (8,10)	58 (17,80)	71 (14,21)	54 (18,94)	**72** (12,88)
3	4 (22,35)	37 (29,40)	31 (37,08)	**39** (31,10)	34 (34,26)
4	293 (9,37)	258 (10,68)	**332** (7,29)	245 (10,86)	303 (9,08)
5	822 (2,60)	826 (2,58)	827 (2,58)	828 (2,58)	**928** (2,35)
6	180 (8,27)	155 (9,67)	**183** (8,12)	174 (8,57)	174 (8,57)
7	17 (8,33)	14 (10,38)	17 (8,33)	12 (10,70)	**18** (7,75)
8	1 (1,92)	5 (22,07)	9 (12,26)	**12** (9,04)	10 (11,04)
9	1 (6,32)	1 (6,22)	2 (4,66)	**16** (5,91)	14 (8,13)
10	234 (1,13)	225 (1,17)	246 (1,07)	235 (1,12)	**402** (0,70)
11	31 (4,67)	33 (4,39)	31 (4,67)	31 (4,67)	**46** (3,43)
12	98 (1,76)	106 (1,63)	109 (1,58)	107 (1,61)	**124** (1,42)

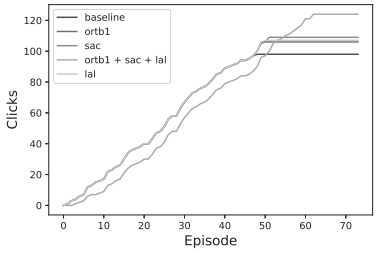

Fig. 6. Ad 12. Win clicks.

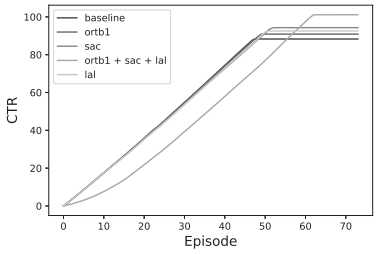

Fig. 7. Ad 12. Agent reward

TABLE VI. WIN IMPRESSIONS AND % OF SPENT BUDGET

Ad id	LinBid	RandBid	DRLB	SAC
1458	37351 (100%)	28024 (100%)	54085 (97,2%)	51810 (97,4%)
2259	16177 (100%)	15388 (100%)	44983 (90,2%)	37822 (90,1%)
2261	20306 (100%)	15346 (100%)	45023 (82,8%)	41434 (90,0%)
2821	44596 (100%)	23679 (100%)	78763 (90,7%)	83071 (100%)
2997	20425 (100%)	8794 (100%)	19583 (96,6%)	42894 (100%)
3358	11383 (100%)	9202 (100%)	12582 (91,3%)	12254 (91,4%)
3386	27672 (100%)	22191 (100%)	36381 (98,0%)	34202 (97,7%)
3427	26117 (100%)	21931 (100%)	29564 (96,7%)	27024 (95,0%)
3476	22793 (100%)	19110 (100%)	26691 (97,8%)	17411 (95,7%)

TABLE VII. IPINYOU DATASET. COMPARISON WITH SOTA. CLICKS AND ECPC

Ad id	LinBid	RLB	CMDP	DRLB	SAC
1458	464 (1,09)	424 (3,09)	464 (2,71)	**465** (3,22)	461 (2,86)
2259	7 (173,52)	12 (101,2)	13 (89,47)	9 (104,9)	**15** (67,33)
2261	9 (105,67)	**11** (87,39)	8 (118,10)	8 (89,2)	9 (92,50)
2821	40 (40,26)	**47** (39)	39 (45,32)	36 (41,8)	40 (34,18)
2997	64 (2,73)	82 (3,7)	71 (2,95)	52 (7,28)	**88** (3,76)
3358	189 (3,77)	199 (4,29)	208 (3,38)	205 (3,36)	**212** (3,33)
3386	55 (5,52)	61 (21,21)	**92** (12,99)	88 (15,2)	89 (14,69)
3427	203 (6,55)	261 (5,14)	292 (4,47)	**296** (4,4)	294 (4,16)
3476	162 (5,92)	131 (9,87)	**181** (7,16)	171 (7,52)	170 (4,72)

of campaigns). This is a positive result, which indicates the effectiveness of the algorithm.

The results of the agent are presented on Fig. IV-E and Fig. IV-E. They show win clicks by the agent for the episodes and the total reward of the agent (CTR).

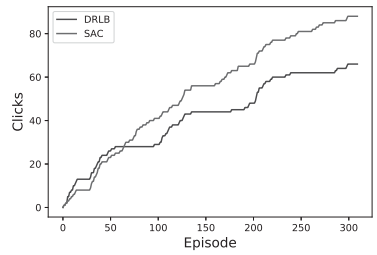

Fig. 8. Ad 2997. Win clicks

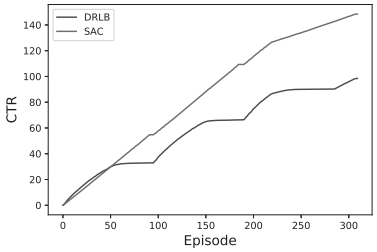

Fig. 9. Ad 2997. Agent reward

Table VIII and Table IX present the final results of the SAC algorithm with all the improvements on the VK anonymized dataset. According to the results of testing on the social network dataset, SAC wins more clicks for 83% of advertising campaigns. On average, the number of clicks earned for a campaign increased by 26%. This is a positive result, which indicates the effectiveness of the algorithm.

F. New ads

The agent does not have any information for new ads and it is difficult to determine the strategy for calculating the dynamic bid.

A common practice for new ads is using exploration strategies of the space. For example, using a greedy strategy or using Thompson's sampling. On the other hand, we can use the cold start strategy for ads, for example, when calculating a bid or click probability for a specific impression, also calculate

TABLE VIII. THE NUMBER OF WINNING IMPRESSIONS

Ad id	LinBid	DRLB	SAC
1	5526	6613	5665
2	7136	2472	5589
3	26574	28730	27163
4	26986	38379	33244
5	88734	193746	199530
6	36233	244861	237780
7	711	808	729
8	496	528	484
9	1073	726	743
10	9820	13014	15208
11	5517	3924	7225
12	24024	23453	23822

TABLE IX. VK DATASET. COMPARISON WITH SOTA. CLICKS AND ECPC

Ad id	LinBid	DRLB	SAC	Improvement
1	82 (9.8)	68 (11.2)	**90** (8.48)	+32%
2	50 (20.6)	71 (7.8)	**72** (12.88)	+1%
3	30 (40.4)	27 (42.5)	**34** (34.26)	+26%
4	248 (6.9)	**331** (8,2)	303 (9.08)	-9%
5	341 (2.1)	836 (2.5)	**928** (2.35)	+11%
6	37 (5.7)	**180** (8.27)	174 (8.57)	-4%
7	14 (11.0)	17 (8.3)	**18** (7.75)	+6%
8	3 (36.8)	6 (18.3)	**10** (11.04)	+66%
9	5 (23.2)	8 (29.0)	**14** (8.13)	+75%
10	314 (0.9)	247 (1.0)	**402** (0.70)	+62%
11	21 (3.3)	33 (4.3)	**46** (3.43)	+39%
12	110 (1.6)	109 (1.5)	**124** (1.42)	+14%

the possible error and calculate the bid taking into account this error.

Let's see what happens if an already trained agent is launched on a new advertising campaign without exploration of the strategy and compare the results of the DRLB and SAC agents. Results are presented in Table X.

TABLE X. NEW ADS. CLICKS AND ECPC

Ad id	LinBid	DRLB	SAC
1458	105 (14.6)	131 (11.4)	454 (1.7)
2259	4 (261.6)	7 (143.2)	4 (133.9)
2261	9 (95.8)	7 (118.1)	5 (87.2)
2821	15 (110.7)	6 (269.6)	20 (36.8)
2997	36 (10.8)	61 (6.3)	76 (3.4)
3358	48 (15.7)	108 (6.7)	82 (8.9)
3386	34 (40.2)	22 (58.8)	61 (17.4)
3427	47 (28.6)	54 (24.1)	26 (50.2)
3476	39 (33.7)	43 (29.3)	95 (13.2)

The Table shows that the agent was able to win clicks much less than without excluding samples of ads from the training set. And this is expected, because each ad has its own distribution of click probabilities, its own distribution of bids, so for correct work if the agent, it needs to have the opportunity for exploring the auction space of this ad.

V. DISCUSSION

The main result of this work is an algorithm for calculating the optimal bid per an impression for an advertising campaign in an online auction in order to maximize campaign targets.

Previous chapter presents the results of each improvement in the algorithm, it shows that each of them contributes to the quality of the agent.

A comparison with the actual published results shows that the algorithm works better than currently published.

The main result will be given in Table IX. According to the test results, the proposed algorithm wins more clicks for 83% of advertising campaigns. On average, the number of clicks earned for a campaign increased by 26%.

In addition, a comparison of the proposed algorithm with alternative approaches, also proposed by the author, is presented, and it shows that the proposed algorithm is optimal relative to the considered alternatives.

The chapter concludes with an analysis of the agents work on new advertising campaigns whose work data is not in the training set. It can be seen that the quality of work is deteriorating, therefore, for the new advertising campaigns, the exploration stage is important, during which positive examples are collected for training the agent.

VI. CONCLUSION AND FUTURE WORK

The main result of this work is an algorithm for calculating the optimal bid for an impression for an advertising campaign in an online auction in order to maximize campaign targets. A novel approach with soft actor-critic and the Lookalike algorithm was used to explore user space was implemented.

Having analyzed existing solutions, we identified the best method at the moment from the article [11], which became the main one for comparison. To measure the quality of the result, three metrics were built: the number of clicks, $eCPC$, CTR as an agents reward.

The algorithm has higher quality comparing to the state-of-the-art methods. According to the experiment results, the proposed algorithm wins more clicks for 83% of advertising campaigns. On average, the number of clicks earned for a campaign increased by 26%.

For future work, we will investigate a model-free approach with Deep Reinforcement learning for bid landscape forecasting and bid optimization into a single optimization framework. In addition, we are planning to explore further how to improve the performance of proposed strategy to handle the highly dynamic environment.

ACKNOWLEDGEMENTS

The paper was supported by the Government of the Russian Federation, Grant 08-08.

REFERENCES

[1] R. Sinha, D. Singal, P. Maneriker, K. Chawla, Y. Shrivastava, D. Pai, and A. R. Sinha, "Forecasting granular audience size for online advertising," *arXiv preprint arXiv:1901.02412*, 2019.

[2] A. Sayedi, K. Jerath, and M. Baghaie, "Exclusive placement in online advertising," *Marketing Science*, vol. 37, no. 6, pp. 970–986, 2018.

[3] J. McDonald, "Regulation, user migration, amazon is the duopolys future assured?" https://www.warc.com/newsandopinion/opinion/regulation_user_migration_amazon_is_the_duopolys_future_assured/3029, 2019, accessed: 2019-09-10.

[4] H. Cai, K. Ren, W. Zhang, K. Malialis, J. Wang, Y. Yu, and D. Guo, "Real-time bidding by reinforcement learning in display advertising," in *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. ACM, 2017, pp. 661–670.

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

[6] C.-r. Zhang and E. Zhang, "Optimized bidding algorithm of real time bidding in online ads auction," in *2014 International Conference on Management Science & Engineering 21th Annual Conference Proceedings*. IEEE, 2014, pp. 33–42.

[7] C. Perlich, B. Dalessandro, R. Hook, O. Stitelman, T. Raeder, and F. Provost, "Bid optimizing and inventory scoring in targeted online advertising," in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2012, pp. 804–812.

[8] W. Zhang, S. Yuan, and J. Wang, "Optimal real-time bidding for display advertising," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2014, pp. 1077–1086.

[9] C. H. Papadimitriou and J. N. Tsitsiklis, "The complexity of markov decision processes," *Mathematics of operations research*, vol. 12, no. 3, pp. 441–450, 1987.

[10] K. Doya, K. Samejima, K.-i. Katagiri, and M. Kawato, "Multiple model-based reinforcement learning," *Neural computation*, vol. 14, no. 6, pp. 1347–1369, 2002.

[11] D. Wu, X. Chen, X. Yang, H. Wang, Q. Tan, X. Zhang, J. Xu, and K. Gai, "Budget constrained bidding by model-free reinforcement learning in display advertising," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM, 2018, pp. 1443–1451.

[12] D. Wu, C. Chen, X. Yang, X. Chen, Q. Tan, J. Xu, and K. Gai, "A multi-agent reinforcement learning method for impression allocation in online display advertising," *arXiv preprint arXiv:1809.03152*, 2018.

[13] J. Jin, C. Song, H. Li, K. Gai, J. Wang, and W. Zhang, "Real-time bidding with multi-agent reinforcement learning in display advertising," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM, 2018, pp. 2193–2201.

[14] G. Micchi, S. Soheily-Khah, and J. Turner, "A new optimization layer for real-time bidding advertising campaigns," *arXiv preprint arXiv:1808.03147*, 2018.

[15] K. Ren, W. Zhang, K. Chang, Y. Rong, Y. Yu, and J. Wang, "Bidding machine: Learning to bid for directly optimizing profits in display advertising," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 4, pp. 645–659, 2017.

[16] G. Jauvion, N. Grislain, P. Dkengne Sielenou, A. Garivier, and S. Gerchinovitz, "Optimization of a ssp's header bidding strategy using thompson sampling," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2018, pp. 425–432.

[17] O. Chapelle and L. Li, "An empirical evaluation of thompson sampling," in *Advances in neural information processing systems*, 2011, pp. 2249–2257.

[18] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," *arXiv preprint arXiv:1801.01290*, 2018.

[19] H. Liao, L. Peng, Z. Liu, and X. Shen, "ipinyou global rtb bidding algorithm competition dataset," in *Proceedings of the Eighth International Workshop on Data Mining for Online Advertising*. ACM, 2014, pp. 1–6.

[20] M. Du, R. Sassioui, G. Varisteas, M. Brorsson, O. Cherkaoui *et al.*, "Improving real-time bidding using a constrained markov decision process," in *International Conference on Advanced Data Mining and Applications*. Springer, 2017, pp. 711–726.