Combining Powers of Two Predictors in Optimizing Real-Time Bidding Strategy under Constrained Budget

Chi-Chun Lin[†*] Kun-Ta Chuang[‡§] Wush Chi-Hsuan Wu[†*] Ming-Syan Chen[†¶]

Department of Electrical Engineering, National Taiwan University, Taiwan, R.O.C.[†]
Department of Computer Science and Information Engineering, National Cheng Kung University, Taiwan, R.O.C.[‡]
{megaacc.lin, wush978}@gmail.com[*] ktchuang@mail.ncku.edu.tw[§] mschen@ntu.edu.tw[¶]

ABSTRACT

We address the bidding strategy design problem faced by a Demand-Side Platform (DSP) in Real-Time Bidding (RTB) advertising. A RTB campaign consists of various parameters and usually a predefined budget. Under the budget constraint of a campaign, designing an optimal strategy for bidding on each impression to acquire as many clicks as possible is a main job of a DSP. State-of-the-art bidding algorithms rely on a single predictor, namely the clickthrough rate (CTR) predictor, to calculate the bidding value for each impression. This provides reasonable performance if the predictor has appropriate accuracy in predicting the probability of user clicking. However when the predictor gives only moderate accuracy, classical algorithms fail to capture optimal results. We improve the situation by accomplishing an additional winning price predictor in the bidding process. In this paper, a method combining powers of two prediction models is proposed, and experiments with real world RTB datasets from benchmarking the new algorithm with a classic CTR-only method are presented. The proposed algorithm performs better with regard to both number of clicks achieved and effective cost per click in many different settings of budget constraints.

1. INTRODUCTION

Online display advertisement (AD) is the main source of revenue for Internet business. After years of evolution, the mechanism of display AD has changed from the pre-allocated style to keyword-based matching of sponsored search, and recently advanced to the per-impression manner of Real-Time Bidding (RTB) paradigm [2]. Via real-time auctions, the RTB solution enables automatic AD impressions selling and purchasing between advertisers and publishers. According to Google's white paper [2], RTB has been promisingly recognized as the leading mechanism for online AD markets. Participants in the RTB ecosystem include publishers, Supply-Side Platforms (SSPs), advertisers, Demand-Side Platforms (DSPs), AD exchanges and networks, and users surfing on the Internet. While a user visits a Web page or activates a mobile app of a publisher with an AD slot available, the attributes of this user and the AD slot are sent via the AD exchange to the DSP. DSP will determine whether the pair, consisting of the user and the slot, conforms to the targeting rules of its customers (advertisers), and decide whether to acquire this impression to display the AD provided by the advertisers in real time. If this impression is of interest, a bidding value is calculated and sent to the AD exchange for auction. Typically, a *second price auction* is held to select the winner. That is, the auction winner, who sent the highest bidding price, will pay the second highest bidding value among all bidders in the auction to the publisher and get her AD displayed. This amount paid by the advertiser is regarded as the *winning price* (WP) of the impression in the auction. Refer to Figure 1 for an illustration of the RTB auction flow. Details of the RTB mechanism can be found in [7].

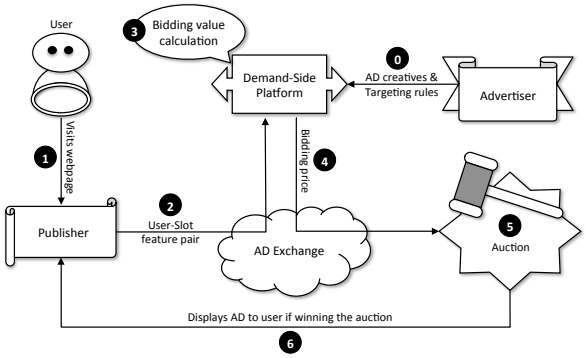


Figure 1: An illustration of the RTB auction flow.

DSP plays an important role on behalf of advertisers in the RTB ecosystem. Its primary task is to determine the bidding price for each incoming AD impression opportunity in real time where the price has to reflect the value of the impression. Typically an impression's value is evaluated by key performance indicators (KPIs) such as clickthrough rate (CTR) or conversion rate (CVR). An impression with a higher expected KPI will be assigned a greater bidding value. There have been works in the literature studying strategies that help the DSP to calculate the bidding price from the expected KPI of an impression. State-of-the-art optimal bidding algorithms rely on a single input, namely the CTR or CVR predictor, to calculate the bidding value. The optimal performance will result from a predictor that correctly estimates the related KPI. If the predictor can merely acquire moderate accuracy, however, it is expected that this kind of solutions will fail to capture optimal results.

In the literature, there have been works regarding the prediction of an impression's winning price from its features. With the winning price becoming directly predictable under reasonable accuracy, we are able to improve the quality of the bidding strategy. Specifically, if an additional winning price predictor is accomplished in the bidding process, the computed bidding value will be able to reflect the true value of an impression more precisely and enhance the overall efficiency for the bidding process of the AD campaign. This sort of framework could provide better results than

CIKM'16 , October 24-28, 2016, Indianapolis, IN, USA

DOI: http://dx.doi.org/10.1145/2983323.2983656

previous models relying on a single KPI predictor. We propose in this paper a new method combining powers of two prediction models, namely CTR and WP predictors. We first describe shortcomings of classical single predictor methods, and then develop our new algorithm. The efficiency of the algorithm is shown by analogy to the well known online stochastic knapsack problem. We also perform experiments to benchmark the proposed method with one of the classic strategies. The experimental results show that our method, when applied on real world RTB datasets, performs better with regard to both number of clicks achieved and effective cost per click in many different settings of budget constraints.

The contributions of this work are multi-fold. (1) We describe the shortcomings of classical bidding schemes and address their non-optimality. (2) We develop a new algorithm that utilizes two prediction models and discuss its optimality. (3) To the best of our knowledge, we are the first to combine two predictors in designing a bidding strategy for RTB. (4) Experiments with real world datasets are performed to assess the efficiency of the new method. We also set up a formal procedure for benchmarking RTB bidding algorithms by adopting the competitive analysis.

2. PRELIMINARIES

2.1 Related Work

Online computational AD is a rapid evolving area with a number of sub-disciplines. Among them, RTB has attracted more and more research interests in recent years. Works in RTB concern various aspects including *bid landscape forecasting* [1], *bid winning price prediction* [8], and *bidding strategy design* [10, 5, 9]. Interested readers can refer to [7] for the comprehensive review.

Two state-of-the-art bidding strategies are discussed in [5, 10] both with the performance measured in terms of number of clicks achieved in the campaign lifetime under a certain budget constraint. In [5], the bidding value is determined by a linear function of the predicted CTR of an impression. The work [10] develops a new function which is in a concave instead of linear relationship with the predicted CTR. It is supposed that the algorithm named ORTB in [10] be optimal if only one prediction model, namely the CTR predictor, can be used. We propose to further increase the performance of the bidding function if one additional estimator, the WP predictor, is incorporated into the bidding value calculation.

The work [4] employs the similar idea of incorporating winning price prediction into the bidding value computation. However, it considers only the winning price predictor while we take into account CTR and WP predictors simultaneously. To sum up, there have been works concerning bidding strategy design on top of the CTR predictor or the WP predictor, but existing ones design their algorithms with either one of the two. We are the first, to the best of our knowledge, who incorporate both of the two predictors in developing an optimal RTB bidding algorithm.

2.2 Problem Formulation

We start by formally stating the problem to be addressed. Note that there are various pricing schemes in RTB and we limit our discussion to CPM (cost per mille) pricing. In light of CPM, the price of winning an auction is immediately charged by the publisher after the advertiser's AD creative is displayed, regardless of whatever the user's action after seeing the AD. Also, the second price auction mechanism is assumed. Given the set of impression opportunities $\psi : \{\vec{x}_1, \vec{x}_2, ..., \vec{x}_n\}$ of a campaign with each impression represented by its *feature vector* $\vec{x}_i | 1 \leq i \leq n$, and the campaign budget B, we wish to obtain as many clicks as possible by bidding on these impressions while keeping the total expense under the budget constraint. We have to decide 1) Whether to place a bid on each $\vec{x}_i$ or not, and 2) If it is to bid, what is the bidding value? We denote the outcome (click or not) of winning $\vec{x}_i$ by an indicator variable $I(\vec{x}_i)$ where $I(\vec{x}_i)$ equals 1 if winning $\vec{x}_i$ leads to a click and 0 otherwise. The exact cost of $\vec{x}_i$ is denoted by $P(\vec{x}_i)$, which means the actual amount to be paid for winning $\vec{x}_i$. That is, $P(\vec{x}_i)$ is the winning price of $\vec{x}_i$. If bidding with $b(\vec{x}_i)$ does not win the auction for $\vec{x}_i$, $I(\vec{x}_i)$ and $P(\vec{x}_i)$ will be both 0. Now the *Bidding Strategy Design* (BSD) problem is defined as follows.

DEFINITION **BSD**: Determine the bidding value $b(\vec{x}_i)$ for each $\vec{x}_i \in \psi$ to maximize $\sum_{i=1}^{n} I(\vec{x}_i)$, subject to $\sum_{i=1}^{n} P(\vec{x}_i) \leq B$.

Note that we answer both questions aforementioned by a single function $b(\cdot)$. Specifically, for the first question, if $b(\vec{x}_i)$ equals 0, we skip this impression and the corresponding $I(\vec{x}_i)$ and $P(\vec{x}_i)$ equal 0 consequently. Also keep in mind that $P(\vec{x}_i)$ is always less than $b(\vec{x}_i)$ since the auction is in a second price manner.

2.3 Challenges in Classical Algorithms

With the BSD problem formally formulated, we present a brief review on two classical algorithms, LIN [5] and ORTB [10], that rely on a single CTR predictor to determine the bidding value for an impression. LIN models an impression's bidding value as a linear function of its predicted CTR (denoted as $pCTR$ hereafter) by

$$b_{LIN}(\vec{x}_i) = \lambda_{LIN} \cdot pCTR(\vec{x}_i). \quad (1)$$

ORTB calculates the bidding value for an impression from a concave-shaped function of its $pCTR$ as[1]

$$b_{ORTB}(\vec{x}_i) = \sqrt{\frac{c}{\lambda_{ORTB}} \cdot pCTR(\vec{x}_i) + c^2} - c. \quad (2)$$

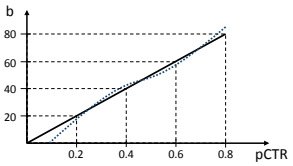
In these two equations, λ_{LIN}, λ_{ORTB} and c are tunable parameters of the corresponding algorithms. Clearly, they both produce the bidding value for an impression from the predicted CTR. LIN is a so-called *truth telling* strategy emphasizing that an impression's bidding value should be linearly proportional to its expected usefulness, while ORTB is also truth telling except that it models the bidding value by a slightly complicated yet still monotonically increasing function of the worth expectation of an impression.

While the corresponding works of LIN and ORTB demonstrated their effectiveness, we now claim and prove that these classical algorithms cannot lead to the optimal result.

THEOREM 1. *Classical BSD algorithms relying on merely a single CTR predictor cannot be optimal.*

PROOF. We prove by giving a counterexample. Consider a campaign with B = 130 and 4 impressions shown in the right of Figure 2 (ignore the pWP and bid-eff. columns for now). Clearly, an optimal algorithm should place a bid on $\vec{W}$ and a bid on $\vec{Y}$ with $b(\vec{W})$ > 80 and $b(\vec{Y})$ > 40 while bidding on $\vec{X}$ and $\vec{Z}$ with $b(\vec{X})$ < 60 and $b(\vec{Z})$ < 20 if impressions can arrive in any order. Such a bidding scheme wins $\vec{W}$ and $\vec{Y}$ with total cost 120 < 130 and obtains 2 clicks without wasting the budget on the non-clicking impressions $\vec{X}$ and $\vec{Z}$. However, neither ORTB nor LIN can achieve this no matter how their parameters are tuned. We plot the bidding/winning price – pCTR relationship for these impressions at the left part of Figure 2. The dotted curve conforms to the optimal bidding strategy, which is in a shape that none of the two algorithms' bidding function can possibly be. It is because the dotted curve corresponds

[1]There is another variant with a more complicated model in [10], but it performs similarly to the one presented here. As such, we omit the discussion of that alternative.



Imp.	Click	pCTR	WP	pWP	bid-eff.
$\vec{W}$	1	0.8	80	82	0.009756
$\vec{X}$	0	0.6	60	62	0.009677
$\vec{Y}$	1	0.4	40	41	0.009756
$\vec{Z}$	0	0.2	20	21	0.009524

Figure 2: An illustrative sample scenario of RTB campaign.

to a polynomial of degree 3, while the degree of ORTB's function is only 2, not to speak of the degree-1 (linear) function of LIN. □

It might be argued that, however, one is able to design a bidding function with a polynomial of degree 3 or greater. But please note that the sample campaign is just an over-simplified scenario consisting of only 4 impressions. If the similar pattern of impressions such as $\vec{Z}, \vec{Y}, \vec{X}$ (ordered by increasing pCTR first, and then the click pattern happens to be $0, 1, 0$) or $\vec{Y}, \vec{X}, \vec{W}$ (click pattern $1, 0, 1$) occurs repeatedly in a campaign (this is inevitable since the CTR predictor has limited accuracy), one will need to design a bidding function of an unrealistic high degree, which sounds meaningless and whose relationship of bidding value versus predicted CTR is difficult to be explained. Hence, we emphasis that in addition to the predicted CTR, other factors should also be considered in designing the bidding strategy in order to deal with the stochastic relationship of the actual click behavior versus the predicted CTR.

2.4 Dataset Overview and Intuition

We use the iPinYou dataset[2] to develop our algorithm and perform experiments. The dataset is a collection of real world RTB DSP logs used in a bidding algorithm competition held in 2013[3]. It consists of 3 seasons' data. For each season, the competition officer released to participants a training set for algorithm development and parameter tuning, as well as a testing set for validation and offline leaderboard benchmarking. A thorough analysis report on this dataset can be found in [11].

According to the argument of [1], the winning bid prices of auctions in RTB follow the log-normal distribution. A trivial extension can be made as: in RTB, the winning bid prices of auctions for impressions *with user clickthrough* also follow the log-normal distribution. Moreover, inspired by [8] that proposed a method for predicting individual impression's winning price directly from its feature vector, we are able to develop a BSD solution utilizing the characteristic of the winning price distribution of a campaign as well as the predicted winning price of each impression in the campaign. The algorithms will be described in the next section.

3. METHODOLOGY

In this section, our BSD algorithms are presented. The algorithms are divided into two phases. The first phase inputs the training dataset and trains the models required in the second phase while the latter phase determines the actual bidding price for each impression in the online bidding process with the testing dataset. We discuss algorithms in two cases: one with perfect CTR and WP prediction, and the other with imperfect prediction. Though the perfect prediction can not possibly be realistic, we still deliberate it because it helps us gain some insight into the development of an algorithm under the condition of imperfect prediction.

3.1 Perfect Prediction: Algorithm PERF

When the CTR and WP predictors are flawless (in other words, having 100% prediction accuracy), what we are concerned about is to spend the budget in the most efficient way so that we can acquire the most clicks with the lowest budget consumption. The algorithm is called PERF and its two phases are described below.

Phase 1 (model training)
Input:
Impression logs
Output:
I - Click predictor
P - Winning price (cost) predictor
Ω - Campaign's PDF of winning prices with click
N_{tr}, N_{te} - Number of impressions with click

The click predictor I is used to forecast whether an impression opportunity will have clickthrough or not. The impression is represented by a feature vector $\vec{x}$ consisting of user attributes (e.g., age, gender, location), AD attributes (e.g., dimension, visibility, title) and attributes from the publisher side (e.g., URL/App name, content category). For some $\vec{x}$, we are sure that $I(\vec{x})$ is either 1 or 0 because it is predicted perfectly. The winning price predictor P reveals the winning price (i.e., cost) $P(\vec{x})$ for $\vec{x}$. For approximation in practice, I can be trained by some well known binary classification methods (e.g., SVM, random forest), and P can be learned by the censored regression proposed in [8]. The distribution PDF Ω can be obtained by simply calculating the average and standard deviation of winning prices from impression logs with clickthrough since we know it conforms to the log-normal distribution, or alternatively by the more sophisticated method described in [1]. The number of impressions with click serves as a basis for estimating the number of clicking impressions from the testing campaign under the assumption that both training and testing sets exhibit similar distributions. We just count the number of impressions with click from the training set, and multiply it by the ratio of campaign lifetime to produce the expected number of clicking impressions of the testing campaign. That is, $N_{te} = N_{tr} \times \frac{T_{te}}{T_{tr}}$ where T_{tr} and T_{te} is the campaign lifetime of training and testing sets respectively. I, P, Ω and N_{te} will be used in the online bidding phase as follows.

Phase 2 (online bidding)
Input:
B - Campaign budget
$\vec{x}$ - Individual impression feature vector
State variable:
B_{cur} - Current budget available, initialized as B
Output:
$b(\vec{x})$ - Bid value for $\vec{x}$ (0 means don't bid on $\vec{x}$)
Procedure:

1. Solve $\int_0^{\beta} p\, N_{te}\, \Omega(p)\, \mathrm{d}p = B$ for β to obtain the value p_{bound}.
2. Input an impression $\vec{x}$. Check if $I(\vec{x}) = 1$. If yes, go to Step 3. Otherwise return 0 and continue with the next impression. Stop if there is no impression left.
3. Use $P(\vec{x})$ to obtain the expected cost for $\vec{x}$.
4. If $P(\vec{x}) \leq B_{cur}$ and $P(\vec{x}) \leq p_{bound}$, return $P(\vec{x})+\delta$ where δ is a small amount (can be the minimum unit of currency accepted in the auction) and update B_{cur} accordingly as Step 5. Otherwise return 0 and go back to Step 2.
5. $B_{cur} := B_{cur} - P(\vec{x})$ if bidding with $P(\vec{x}) + \delta$ on $\vec{x}$ wins the auction. If B_{cur} becomes 0, or there is no impression left, then stop. Otherwise go back to Step 2.

Here are details of the design. With the perfect click predictor, we know whether an impression will lead to a click or not, thus we skip all impressions with $I(\vec{x}) = 0$ and proceed further only with

[2]Downloaded from http://data.computational-advertising.org
[3]See http://contest.ipinyou.com

those having $I(\vec{x}) = 1$ in Step 2. For each clicking impressions, we know its exact cost by the perfect WP predictor $P(\vec{x})$. To allocate the budget to acquire as many clicks as possible, we begin spending on impressions with the cheapest price and proceed to those with higher prices gradually until all budget is consumed, or until the campaign has passed all over. The integral in Step 1 and the comparison of $P(\vec{x})$ with p_{bound} in Step 3 reflects this strategy.

3.2 Imperfect Prediction: Algorithm PRUD

In this subsection we discuss the case when the CTR and WP predictions are imperfect since it is impossible to have perfect predictors in reality. The algorithm is called PRUD (naming inspired by "a *prudent* bidder") and also consists of two phases.

Phase 1 (model training)
Input:
Impression logs
Output:
$pCTR$ - CTR predictor
pWP - Winning price (cost) predictor
ρ_{cut} - bid efficiency cutoff value

The training phase has three outputs, two of which are CTR predictor and WP predictor. To train the CTR predictor, we refer to the logistic regression proposed in [6]. The censored regression [8] is used in WP model training. In addition, a *bid efficiency cutoff* value is produced. The bid efficiency ρ of an impression $\vec{x}$ is defined as:

$$\rho(\vec{x}) = pCTR(\vec{x}) \,/\, pWP(\vec{x}). \tag{3}$$

The bid efficiency corresponds to the following argument. When an impression is with a high CTR, we would have a strong intention to bid on it because the expected revenue is high due to the high clickthrough rate (i.e., probability of click). However, when an impression has a relatively low CTR but is accompanied with a low predicted WP, it could still be reasonable to "invest" on this impression since even if it ends up with no clickthrough eventually, the loss of investment is insignificant due to the low price (cost). That is, we use the predicted CTR and WP simultaneously to determine whether an impression is worthy of spending our budget. The *higher CTR* and *lower WP* an impression has (thus *bigger* ρ), the more valuable it appears. Therefore, we define the bid efficiency as dividing CTR by WP to reflect this intuition.

After defining the bid efficiency for an impression, we now face a problem: to or not to bid on an impression with a certain bid efficiency value? To remedy it, we use the training dataset to perform a search for a *cutoff* value, that is, the lowest ρ value acceptable for this campaign under a predefined budget constraint. When an impression has a ρ higher than or equal to the cutoff value, we bid on it with the predicted WP plus a small lift-up value (since it is a second price auction). Otherwise the impression is skipped. We find the cutoff value that results in the most clicks achieved in a campaign's training period under some budget constraint. And then, the same cutoff value is used as the threshold in the same campaign's testing period with the same condition of budget constraint.

We now present the online bidding algorithm as follows.

Phase 2 (online bidding)
Input:
B - Campaign budget
$\vec{x}$ - Individual impression feature vector
State variable:
B_{cur} - Current budget available, initialized as B
Output:
$b(\vec{x})$ - Bid value for $\vec{x}$ (0 means don't bid on $\vec{x}$)
Procedure:

1. Input an impression $\vec{x}$. Check if $\rho(\vec{x}) \geq \rho_{cut}$. If yes, go to Step 2. Otherwise return 0 and continue with the next impression. Stop if there is no impression left.
2. If $pWP(\vec{x}) \leq B_{cur}$, return $pWP(\vec{x}) + \delta$ where δ is a small amount (can be the minimum unit of currency accepted in the auction) and update B_{cur} accordingly as Step 3. Otherwise return 0 and go back to Step 1.
3. $B_{cur} := B_{cur} - P(\vec{x})$ if bidding with $pWP(\vec{x}) + \delta$ on $\vec{x}$ wins the auction and its exact winning price is $P(\vec{x})$. If B_{cur} becomes 0, or there is no impression left, then stop. Otherwise go back to Step 1.

An illustrative example for PRUD. Recall the sample campaign in Figure 2. The predicted winning prices for the four impressions and their corresponding bid efficiencies are listed at the pWP and bid-eff. columns respectively. Setting ρ_{cut} to 0.0097, PRUD picks up $\vec{W}$ and $\vec{Y}$, discards $\vec{X}$ and $\vec{Z}$, and produces the optimal result of 2 clicks with budget consumption 120.

3.3 PRUD and Knapsack Problem

It is worth noting that we can relate BSD to the well known *knapsack problem* (KP). We regard each bidding impression opportunity $\vec{x}_i$ as a stone with value v_i and weight w_i. We have a knapsack capable of holding stones and its capacity is W. Our goal is to load the knapsack with as valuable stones as possible while keeping the total weight under the knapsack capacity. For the analogy of PRUD to KP, the value v_i of a stone $\vec{x}_i$ is the corresponding CTR prediction $pCTR(\vec{x}_i)$ and the stone's weight w_i is the associated predicted cost $pWP(\vec{x}_i)$. The campaign budget B is considered as the knapsack capacity W. There are many forms of KP and here we are dealing with the *online stochastic* variant. Specifically, we do not know the characteristic portfolio of all stones in advance. Instead, the stones are coming one after another thus we can inspect a stone's value and weight only after receiving one, and then decide whether to pack it or not immediately on the fly. Once a stone has been packed, we are unable to discard it afterwards. Nor can we reconsider a previously unpacked stone later again. The book [3] provides excellent discussion and algorithms for various types of KP problems. For the online stochastic one, an algorithm named *THRESHOLD* is given as follows. First, an *efficiency threshold* T is defined. Then upon receiving a stone $\vec{x}_i$, its efficiency is calculated by v_i/w_i and compared with T. If the efficiency is greater than or equal to T, the stone is packed. Otherwise the stone is discarded. Each stone is handled sequentially in the coming order until finally the knapsack capacity is exhausted or all stones have been processed. Unfortunately there is no guaranteed optimal solution for the online stochastic KP. The THRESHOLD algorithm is a semi-optimal one if the threshold value T is carefully determined. In our BSD context, algorithm PRUD learns the best threshold value by tuning the bid efficiency cutoff value with the training dataset. Provided the testing dataset has a distribution of impressions similar to the training dataset, and the budget is set to a fixed percentage of the sum of all impressions' cost (say, 1/16 of $\sum P(\vec{x}_i)$), the same cutoff value learned from the training phase can be used in the online bidding phase to produce the same level of semi-optimal result.

4. EXPERIMENTS AND BENCHMARKING

We depict in this section our experiments for benchmarking the two algorithms PRUD and ORTB. We use the 9 campaigns in seasons 2 and 3 of the iPinYou dataset for the experiments because

Season	Cpg.	CTR AUC	WP_{pred} Avg.	WP_{real} Avg.	WP_{pred} RMSE	WP_{real} Stdev.	1/64 ORTB	1/64 PRUD	1/32 ORTB	1/32 PRUD	1/16 ORTB	1/16 PRUD	1/8 ORTB	1/8 PRUD	1/4 ORTB	1/4 PRUD	1/2 ORTB	1/2 PRUD
3	2259	0.8870	102.81	126.64	77.16	79.44	16.67	30.21	29.17	38.54	42.71	51.04	55.21	65.63	69.79	78.13	88.54	95.83
3	2261	0.8756	94.53	123.74	78.79	83.53	28.05	32.93	31.71	42.68	42.68	53.66	53.66	65.85	74.39	79.27	90.24	95.12
3	2821	0.8195	99.39	120.90	66.40	80.29	20.00	21.72	24.14	28.28	33.79	38.62	45.86	51.72	62.76	70.00	81.03	86.21
3	2997	0.7440	69.96	87.36	62.16	75.18	17.04	17.71	23.32	22.65	31.84	31.39	43.72	42.38	57.85	58.52	76.91	78.03
2	1458	0.9257	81.83	91.23	53.18	62.55	72.08	72.30	73.97	73.41	77.42	76.53	80.76	80.53	86.76	85.54	92.66	92.66
2	3358	0.9286	121.54	134.07	50.83	65.42	62.75	66.97	67.34	69.72	72.29	74.13	79.45	80.37	87.16	87.52	92.66	93.03
2	3386	0.8094	111.45	122.46	58.57	69.97	24.78	25.65	29.37	29.12	35.81	36.06	45.85	46.84	59.98	65.18	79.93	80.67
2	3427	0.9025	106.01	116.10	51.44	62.73	59.49	60.70	62.74	63.41	70.73	71.54	75.75	76.02	82.38	81.30	90.65	90.24
2	3476	0.8671	95.69	108.84	53.28	62.97	43.63	47.03	51.27	53.54	60.91	62.61	66.57	68.56	79.60	79.04	88.39	88.39

Table 1: Leftmost: Statistics of CTR and WP predictors. Rightmost: Benchmark results of ORTB versus PRUD in competitive ratio.

they provide a more comprehensive set of features than season 1. We first introduce the preparation of data, followed by some issues of predictor model training and algorithm tuning. Finally, the benchmark results are presented.

4.1 Implementation and Tuning

Data Preparation. It is known that PRUD performs best when training and testing sets express similar impression distributions. ORTB also tunes its parameters with the training set and operates in the testing phase with what learned, so that the same argument (best performance under similar training/testing distributions) holds naturally. Hence, we want to benchmark them under this condition. As such, we randomly sample 2/3 of impressions from the original iPinYou training set for each of the 9 campaigns to produce 9 new data collections which serve as our new training sets, and 1/3 of original training impressions to form the 9 new testing sets.

Feature Extraction. The predictor training requires a feature set as input. The iPinYou dataset provides 24 features and we select 17 of them. All features are extracted and converted to binary by dummy encoding. The number of converted features ranges from 87,845 for the smallest campaign to 510,357 for the largest one.

Predictor Quality. After feature extraction, CTR and WP predictors are trained respectively with L2-regularized logistic regression and censored regression. We list AUC (area under ROC curve) of CTR predictor and RMSE (root mean squared error) of WP predictor in the left of Table 1. The standard deviations of real winning prices and the average values for both predicted and real winning prices are depicted as well. These WP-related statistics are measured from impressions with click, excluding non-clicking ones. From the table, we see that CTR predictor performs well on season 2 but somewhat unsatisfactorily on season 3. Meanwhile, the RMSE values of WP predictor all fall within one standard deviation of real winning prices. It can be observed that the predictor has acceptable (but far from perfect) accuracy in predicting winning prices. Note that predicting an individual impression's winning price from its feature vector under RTB context is still a pioneer subject in the literature, and what we can do best currently is the model proposed by [8] which produces the moderate performance shown here. We will demonstrate that, however, incorporating such an ordinary predictor into algorithm PRUD adds considerable power in optimizing the bidding efficiency.

WP Predictor Lift Value. From Table 1, we see that WP predictor tends to underestimate[4] hence a proper *lift* value has to be added. Otherwise we will lose too many impressions if we just bid according to the original predicted prices. What follows is, how should the lift value be determined? It is difficult to be derived theoretically and after some experiments we decide to find the value empirically by a heuristic procedure. Specifically, we find the smallest L with the training data by the following inequality[5] for each campaign:

$$pWP(\vec{x}) + L > P(\vec{x}) \quad \text{for 95\% of} \quad \{\vec{x} \,|\, I(\vec{x}) = 1\}. \quad (4)$$

After L is determined, we use $pWP'(\vec{x}) = pWP(\vec{x}) + L$ as the new predicted WP for each $\vec{x}$ of the campaign. The lift value not only serves as a compensation to the underestimated winning price, but also acts as the δ value described in the online bidding phase of algorithm PRUD in Section 3.2. We regard L as another parameter of PRUD in addition to the bid efficiency cutoff value ρ_{cut}.

Tuning for ORTB. ORTB has two parameters: c and λ. We follow the ORTB work to tune them. Please refer to [10] for details.

Tuning for PRUD. PRUD also has two parameters. We have explained how the lift value L is determined. What remains is the bid efficiency cutoff value ρ_{cut}. Following the same scenario in [10], we set the budget as 1/64, 1/32, 1/16, 1/8, 1/4 and 1/2 respectively of the total cost of all impressions for each campaign. Since there are 9 campaigns, we have thus a total of $9 \times 6 = 54$ test cases. We use the training set to find the optimal ρ_{cut} that produces the largest number of clicks for each test case.

Tuning for PERF. Though PERF is not realizable in practice, we are still able to implement it because we have the complete testing data on hand. Recall the model training phase from Section 3.1. The click predictor, cost predictor and campaign PDF of winning prices with click (actually we use the exact composition of clicking impressions instead of the probability distribution) are readily revealed by scanning the whole testing data. We implement PERF and use it as the basis for comparing the relative performance of ORTB with PRUD to be discussed in the next subsection.

4.2 Benchmark Results

The results collected from applying each of the three algorithms on the prepared dataset are depicted in the right of Table 1. We follow the convention of *competitive analysis* and present the *competitive ratio* (CR) in the table. For each campaign-budget setting, we run PERF, ORTB and PRUD separately and then calculate the competitive ratio by dividing the number of clicks that ORTB or PRUD delivers by the number of clicks achieved by PERF:

$$CR = \frac{\#Click_{ORTB|PRUD}}{\#Click_{PERF}} \times 100\%. \quad (5)$$

PERF provides the upper-bound a bidding algorithm is able to reach and the competitive ratio indicates how well ORTB or PRUD can do compared to PERF. It can be observed that PRUD wins ORTB in 41 cases, loses to ORTB in 11 cases and ties with ORTB in 2 cases out of the total 54 cases. To ease the assessment, we also draw the CR chart by bar plot in Figure 3.

[4] This is expected as what has been discussed in [8].

[5] The percentage 95% is selected because it performs best among several values tested. 100% is not used because it will result in an extremely large lift value.

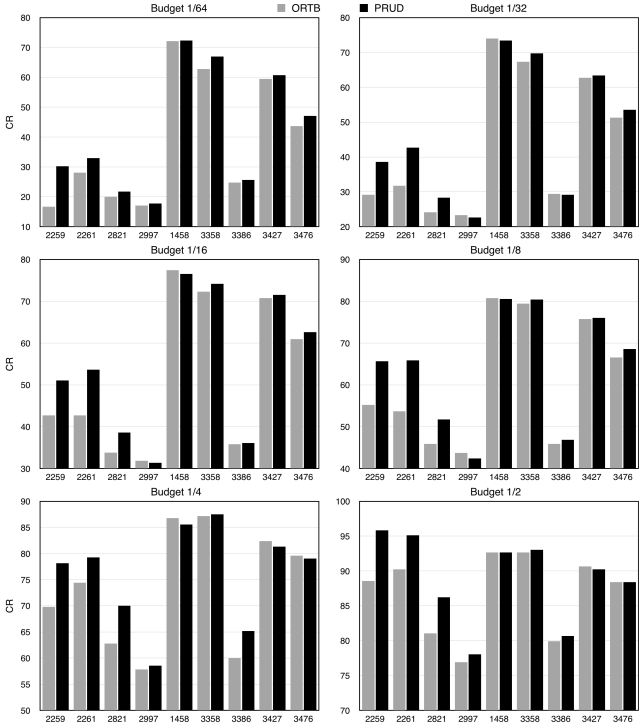


Figure 3: ORTB versus PRUD measured in competitive ratio.

Relation of AUC to algorithm performance. We find in general that, the lower AUC the CTR predictor for a campaign, the more improvement PRUD provides over ORTB. It is because that ORTB relies on merely the CTR predictor and thus a better prediction performance directly results in the delicacy of the final outcome of ORTB. On the other hand, PRUD incorporates additionally the WP predictor, and the WP predictor in some manner compensates the defect of the CTR predictor.

A plus in addition to clicks. We also show the performance comparison by *effective cost per click* (eCPC) in Figure 4. The eCPC is calculated by dividing the total cost of a campaign by the number of clicks obtained. The lower the eCPC is, the more cost-efficiency an algorithm provides. It can be seen that PRUD exhibits lower eCPC than ORTB in almost all cases (51 of 54, or 94%). Even for many cases that PRUD loses to ORTB in terms of number of clicks, PRUD still performs better if measured by eCPC. This is not surprising, though. Since we develop PRUD by starting from pursuing impressions with high bid efficiency while the efficiency is determined by dividing the predicted CTR (related to number of clicks) by the predicted winning price (related to cost), and this formula is in the reciprocal form of eCPC, PRUD favors eCPC (higher efficiency implying lower eCPC) naturally.

5. CONCLUDING REMARKS

In this paper, we develop a new bidding strategy based on two predictors, the CTR and WP predictors, to serve as the bidding value generator for a DSP running its real time auction in the RTB context. The algorithm, named PRUD, is expected to be optimal by analogy to the well known online stochastic knapsack problem. We perform experiments with real world RTB datasets and use a formal competitive analysis benchmarking flow to compare PRUD with the state-of-the-art single-predictor ORTB bidding algorithm. Our results show the outstanding bidding efficiency of PRUD in terms of not only a greater number of clicks achieved, but also a lower effective cost per click for an advertising campaign. By combining

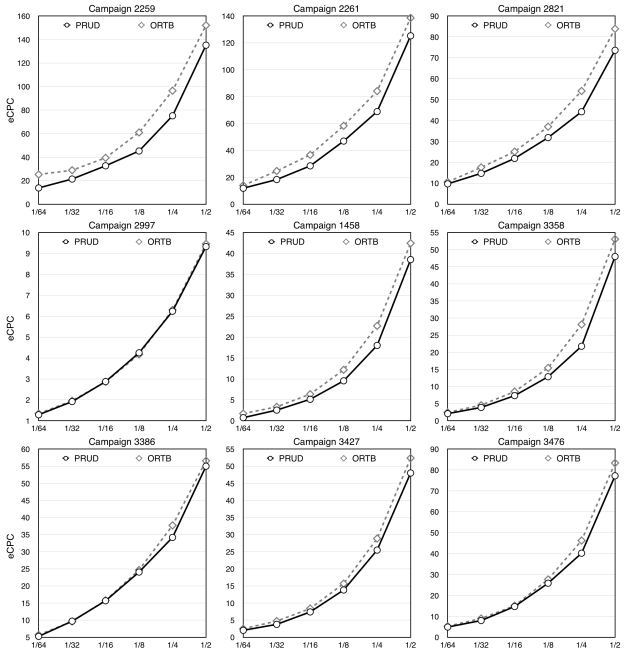


Figure 4: ORTB versus PRUD measured in effective cost per click.

powers of the two predictors, PRUD provides a new cost-efficient strategy for a DSP competing in the rapid emerging RTB market.

Acknowledgments

This paper was supported in part by Ministry of Science and Technology, R.O.C., under Contract 105-2221-E-006-140-MY2, 105-2634-B-006-001, and 104-2221-E-002-214-MY3. The authors also thank Bridgewell for providing their RTB logs for a preliminary analysis in this work.

6. REFERENCES

[1] Y. Cui, R. Zhang, W. Li, and J. Mao. Bid landscape forecasting in online ad exchange marketplace. In *KDD*, 2011.

[2] Google. *The arrival of real-time bidding and what it means for media buyers*. Google, 2011.

[3] H. Kellerer, U. Pferschy, and D. Pisinger. *Knapsack problems*. Springer, 2004.

[4] X. Li and D. Guan. Programmatic buying bidding strategies with win rate and winning price estimation in real time mobile advertising. In *PAKDD*. Springer, 2014.

[5] C. Perlich, B. Dalessandro, R. Hook, O. Stitelman, T. Raeder, and F. Provost. Bid optimizing and inventory scoring in targeted online advertising. In *KDD*, 2012.

[6] M. Richardson, E. Dominowska, and R. Ragno. Predicting clicks: estimating the click-through rate for new ads. In *WWW*, 2007.

[7] J. Wang and S. Yuan. Real-time bidding: A new frontier of computational advertising research. In *WSDM*, 2015.

[8] W. C.-H. Wu, M.-Y. Yeh, and M.-S. Chen. Predicting winning price in real time bidding with censored data. In *KDD*, 2015.

[9] W. Zhang, Y. Rong, J. Wang, T. Zhu, and X. Wang. Feedback control of real-time display advertising. In *WSDM*, 2016.

[10] W. Zhang, S. Yuan, and J. Wang. Optimal real-time bidding for display advertising. In *KDD*, 2014.

[11] W. Zhang, S. Yuan, J. Wang, and X. Shen. Real-time bidding benchmarking with ipinyou dataset. *arXiv:1407.7073*, 2014.