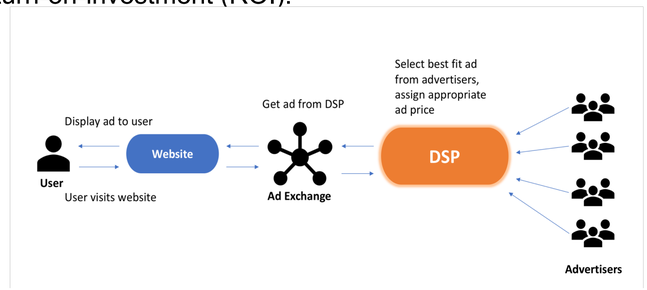
Apply Reinforcement Learning in Ads Bidding Optimization

Ying Chen (SCPD: ychen107)

Background

Online display advertising is a marketing paradigm utilizing the Internet to show advertisements to targeted audience and drive user engagement. Since around 2009 Real-time bidding (RTB) has become popular in online display advertising. RTB allows an advertiser to use computer algorithms to bid in real-time for each individual opportunity to show ads. These bids can be based on the impression-level features, such as user, ads and context information. With its fine-grained user targeting and auction mechanism, RTB has significantly improved the campaign return-on-investment (ROI).



Looking at the process we can see that bidding optimization is one of the most critical problems for the advertiser, which aims to set right bidding price for each auctioned impression to maximize key performance indicator (KPI) such as click.
Most of existing work only focused on finding optimal bid price but ignored an important part of optimal bidding strategy: pacing algorithm. Pacing algorithm is essentially for budget allocation which aims to smooth budget spending across time according to traffic intensity fluctuation.

Methods

We propose to use reinforcement learning algorithms to find the optimal bidding strategy, the major difference of our algorithm from previous work is that our algorithm try to generate bid and pacing signal at the same time. This requires the algorithm to not only learn to adapt to the changes in bidding environment but also the interaction between bid price and pacing signals.
We first focus on intra-day bidding optimization. Given a certain amount of budget and unknown traffic distribution throughout the day, we want to find out a bidding and pacing strategy that maximize the total number of clicks, this is equivalent to minimizing CPC (cost per click).

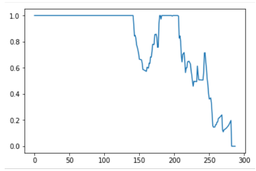
Methods

State space: the state of the campaign is characterized by its remaining budget, remaining delivery time, current value of CPC. To simplify the discussion, we only consider the case when CPC optimization goal is set, which is the most common case.
We define state as s_t = (Budget_t, CPC_t, Hour_t).
Action space: there are two action signals in our setup, the pacing signal p_t, and the Bid adjustment signal a_t
Ideally the pacing signal should be continuous. To reduce algorithm complexity, we discretize this signal in Q-learning. Pacing granularity could be a concern but for the first version it suffices to discretize [0,1] by 2% interval, so we end up with 50 possible values. The bid adjustment signal a_t can be [+/-1%, +/-2%, +/-5%, +/-10%, +/-20%, +/-50% and +100%] on top of previous bid.
State transition: We model the bidding process as a Markov Decision Process, given a previous bid b_t ,bid adjustment signal at and pacing signal p_t, the bidding agent join auctions and observe the outcome, then update the state s_t. Note that we don't know the state transition dynamics, the optimal control policy will be learned using reinforcement learning.
Reward function: The reward function is composed of two parts: the discounted reward and the regularization. The instant reward r_t could be the number of clicks in given time interval t, and a regularization is put on the bid signal to encourage its smoothness.

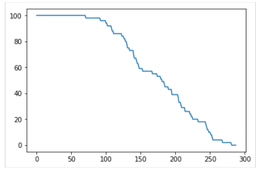
Experiment Result

1. Baseline

We use fixed bidding price and pacing signal as a baseline. It's a simple feedback loop control. We can see the pacing rate has changed a lot during the procedure. And the spending looks reasonable.



Pacing rate in naïve bidding



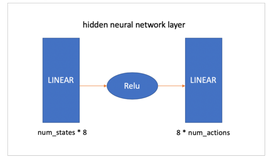
Remaining budget in naïve bidding

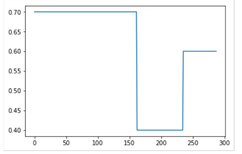
2. DQN

Q-learning is a straightforward off-policy learning algorithm, it basically builds a Q-table which gives the reward function for (state, action) pairs, and update it while exploring the environment. Deep Q-learning is just a neural-network version of Q-learning which uses DNN/CNN to approximate the Q-function.

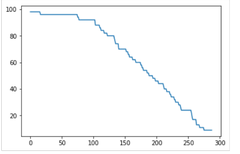
Experiment Result

Here we trained DQN model in 50, 200, 500, 2000 and 5000 iterations. Here is one of the 500_iter results. We can see the pacing rate is relatively smooth and the spending is relatively better.





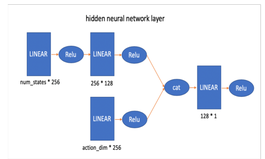
Pacing rate in DQN

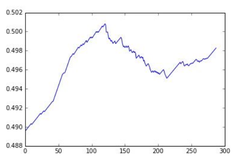


Remaining budget in DQN

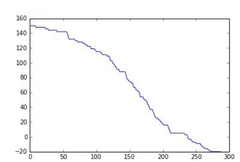
3. DDPG

We also tried DDPG. Comparing with DQN, its action space is continuous. The results look very random and the budget some times overspends.





Pacing rate in DDPG



Remaining budget in DDPG

Conclusion and future work

Since we generate a random amount of impressions in each time slot to mimic the actual traffic, the environment input is entirely different every time. We do not use the quantitative metric to measure the performance of different algorithms. Instead, we compare the budget spending, pacing rate smoothness, and stables among different algorithm in a qualitative way.
Overall, reinforcement learning(DQN) on pacing can spend most of the budget without overspending and has a relatively smooth pacing rate. It shows better performance compared with the baseline algorithm. There are also some weakness in DDQN for now. 1. The training model is not very stable. 2. Sometimes it spends too much money so that the remaining daily budget is negative at the end.
These results give us the confidence to apply Reinforcement Learning algorithm in Bidding Optimization in the Ads industry. In the future, we will modify neural network architecture, refine cost functions and tune the parameters to mitigate the disadvantages. We are also going to explore reinforcement learning in intra-day bidding challenge.

Email: yingchen107@gmail.com