Optimising Contextual Advertising Through Real-Time Bidding With Budget Constraints*

Jingwen Cai
Department Computer Science, Umeå University
Umeå, Sweden
jingwenc@cs.umu.se

Johanna Björklund
Department Computer Science, Umeå University
Umeå, Sweden
johanna@cs.umu.se

Abstract

Online advertising opportunities are bought and sold in automated auctions driven by real-time bidding. In the case of contextual advertising, the size of a bid is informed by the media context in which the ad will be displayed. In contrast to personalised advertising, contextual advertising is better aligned with privacy acts such as GDPR and CCPA. We investigate how reinforcement learning with human feedback can help optimise contextual advertising under budget constraints. We propose a dynamic epsilon-greedy algorithm that considers the rate of budget consumption during a finite transaction time. The goal is to maximise long-term rewards in a sustainable manner. Our comparative evaluation of fundamental reinforcement learning algorithms on real data suggests that the approach is feasible and effective.

CCS Concepts

• **Computing methodologies** → *Reinforcement learning*; • **Information systems** → *Computational advertising*; *Recommender systems*.

Keywords

real-time bidding, exploration and exploitation, budget constraints

1 Introduction

In the context of online advertising, an impression is an opportunity to show an ad to a user. The vast majority of all impressions are sold to advertisers or their intermediates through a mode of automated auctioning known as real-time bidding (RTB). This allows for allocation management and placement of ads with minimal human mediation [9]. The basic RTB interaction scheme looks as follows: When a user visits a webpage, the publishers controlling the page send a bid request through the supply-side platform to an ad exchange. In response, one or more demand-side platforms bid on behalf of different advertisers. Ideally, an optimal bidding strategy uses a small amount of budget to bid on the most valuable impressions and the values of the impressions often hinge on the personal information of the user, which helps advertisers effectively identify their target audience.

With the enactment of a series of data protection policies such as the California Consumer Privacy Act [1] and the General Data Protection Regulation (GDPR) [3], people's awareness of their online privacy is increasing. Data platforms must now explicitly ask for user's consent to collect personal information and track their activities. However, since it is impossible to predict who will win the impressions in advance, the users cannot be given a complete picture of how their data will ultimately be used [9]. An alternative solution is contextual advertising, which does not depend on private data. Instead, the value of an impression is derived from the media context in which the winning ad will appear. The assessment of the context can, for example, be based on the degree of topical congruence between the ad and the context. By studying the leverage of the contextual features under budget constraints, an effective and sustainable advertising strategy can be found.

RTB optimisation is a central topic in advertising literature. To maximise the impact of advertising within budget constraints, accurately predicting click-through rates (CTR) is crucial. Previous studies have largely treated CTR prediction as a static problem and resulted in algorithms such as LIN [7] and ORTB [11] which use linear and nonlinear regression, respectively. There have also been attempts at combining neural networks and factorisation machines to improve CTR prediction [4, 5]. In real life, the processes of recommending and bidding on advertisements are all dynamic in nature. For this reason, RTB is more accurately modelled as a dynamic optimisation problem. The algorithms RLB [2] and DRLB [10] aim to address such dynamic bidding problems with budget constraints. While DRLB addresses the limitation of RLB's model-based approach, making it more applicable to real-world scenarios using Deep Q-Network, the fixed set of bid scaling parameters that spans its action space still restricts its adaptability. Therefore, we reframe RTB as a dynamic optimisation problem and train a bidding agent to spend a limited budget as effectively as possible, to maximise long-term returns in a more sustainable manner.

This paper explores the application of reinforcement learning to contextual advertising with budget constraints. The main contributions are: (1) A dynamic epsilon-greedy reinforcement learning mechanism that adaptively adjusts the rate of advertising spend and maximises long-term rewards. (2) Realistic parameter values, derived from real-world data, including costs and rewards. (3) Offline evaluations on real-world data, under a holistic simulation workflow that combines RTB, contextual advertising, and reinforcement learning.

*Copyright 2024 for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).
Presented at the SURE workshop held in conjunction with the 18th ACM Conference on Recommender Systems (RecSys), 2024, in Bari, Italy.

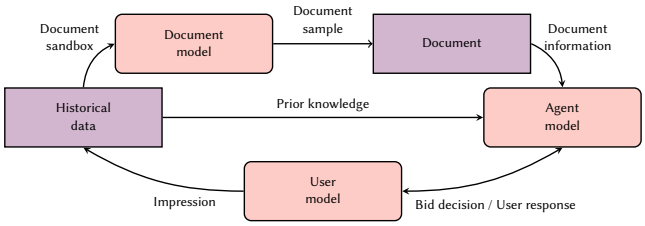


Figure 1: The interaction process revolves around the document, user, and agent models (peach-coloured boxes).

2 Method

In our experiments, we use real-world data to simulate RTB auctions. The dataset consists of pairs of impressions and information about whether the user clicked the ad, i.e., the ground truth that the agent is trying to predict. To train the agent, it is shown the impression part of one data item at a time and has to decide whether or not to bid. It bases its decision on contextual features of the impression, as well as on prior knowledge from previous attempts at bidding. If it chooses to bid and, according to the historical data, the user clicked on the ad, then it receives a positive reward. This bidding process is then repeated until the agent has depleted its budget, or a maximum number of bidding opportunities have been offered. This means that the agent needs to assess not only the value of the impression currently at hand, but also how to maximise the cumulative return of the entire episode with a limited budget. To this end, it must ration its budget over time, rather than bidding on every opportunity which would exhaust the budget prematurely, or being too conservative and eventually running out of bidding opportunities. This reflects real-world applications, where campaigns are set to run over periods of days, weeks, or even months, so it is not desirable that the agent spends its entire budget in the first few minutes.

2.1 Interaction Framework

There are three main components in our simulated bidding process, inspired by RecSim [6]: a document model, a user model, and an agent model. The agent receives a stream of bid requests, each of them generated by sampling the document model to retrieve a candidate advertisement characterised by some contextual feature. For each such document it decides whether or not to bid. If it decides to bid, it is informed if the user clicks on the ad. The agent then computes its gains and losses and updates its bidding strategy. The interaction is illustrated in Figure 1. The training data consists of recorded RTB auctions of individual impressions[1], aggregated from multiple US publisher sites in the period June 14 – 28, 2016. Each impression is represented by a reference ID, a topic ID, and a binary user response: 1 means that they clicked, and 0 that they did not.

[1]https://www.kaggle.com/competitions/outbrain-click-prediction/data

2.2 Optimisation Task

We model the interaction between the agent and the external environment (that is, the user model and the document model) at each time step as a Markov Decision Process (MDP). We denote by a_t the bidding action taken by the agent at time step t, by s_t the state of the environment at that time, and by r_{t+1} the immediate reward received at time step $t+1$. Each time an agent makes a bidding decision, it triggers a subsequent change in the environment state, and the updated state is evaluated by the agent in preparation for the next time step. In our simulated bidding process, each episode is an ordered sequence of several such interactions. We thus obtain the following sequential trajectory at the end of an episode with t time steps: $s_0,\ a_0,\ r_1,\ s_1,\ a_1,\ r_2, \ldots, s_{t-1},\ a_{t-1},\ r_t$. The goal of the agent is to maximise the cumulative return $G = \Sigma_{i\in\{1,\ldots,t\}} r_t$ under a constrained budget. The central concepts in the chosen approach are listed below, followed by an explanation of our choice of parameter values.

- A **state** is a tuple $s = \langle tpc, b\rangle$, consisting of contextual features that describe the essential information of the external environment, where *tpc* is a topic ID of the document and b is a figure denoting the remaining budget. The total budget for each episode in the experiments is \$120, and if the agent chooses to bid, then a bidding price of \$1.5 is deduced from the available budget only if it receives a click.
- An **action** $a_t \in A = \{0, 1\}$ represents the agent's bidding decision at time step t.
- A **reward** r_t from the transition $s_t \to s_{t+1}$. In the experiments, if the agent chooses to bid and receives a click from the user, then the reward is \$5. In all other cases, there is no reward.
- The **episode length** is a non-zero integer parameter t. In the experiments presented here, we let $t = 500$. In the extreme scenario where the agent chooses to bid on almost every impression, the episode will end around $t = 80$ when the budget is exhausted. However, we would like to encourage the agent to be more selective with its bids, and learn how to identify impressions that are likely to lead to clicks.

To make our experiments as realistic as possible, we derive key parameters from Amazon's report on Advertising Cost of Sale (ACoS)[2]. ACoS is a measure of the ratio of advertising expenditure to revenue and should be around 20% – 30%, calculated by:

$$\text{ACoS} = \frac{\text{TotalAdSpend}}{\text{TotalSales}} = \frac{\text{CPC}}{\text{UnitPrice} \times \text{CVR}}$$

Here, CVR denotes the conversion rate, that is, the relative proportion of users that click on an ad that also take some desired action when arriving at the landing site. The Cost Per Click (CPC) is generally 70 – 150 cents, and CVR should be in the range of 2% – 8%.

[2]https://advertising.amazon.com/library/guides/acos-advertising-cost-of-sales

2.3 Learning Algorithms

We compare two fundamental classes of RL algorithms: the Monte Carlo (MC) family and the Temporal Difference Learning (TD) family. They are represented by first-visit MC, every-visit MC, Q-Learning, and SARSA [8]. As previously mentioned, the agent evaluates whether to bid based on the state, and the different actions result in different feedback. Thus, for each state-action pair, we obtain a score to measure its value, denoted as $q\langle s_t, a_t\rangle$. Once our agent has taken the action a_t at the state s_t and received a reward, the value $q\langle s_t, a_t\rangle$ can be calculated based on the Bellman equation [8] and used for future policy guidance.

One of the most important problems in reinforcement learning is to balance exploration and exploitation. Exploitation is when the agent selects the option it expects to yield the highest reward based on the information it has collected thus far. Exploration, on the other hand, is when the agent ignores previous knowledge to investigate unknown (and potentially highly rewarding) options. In the real world, users are not a heterogeneous group, and even individual users change their behaviours over time and in response to external factors. So, if the agent is only guided by prior experiences, it is likely to overlook opportunities for higher rewards and become trapped in sub-optimal choices. In contrast, if the agent varies its interaction with the environment, it gains a more comprehensive understanding of the underlying dynamics.

As previously mentioned, we want the agent to economise with its budget throughout the episode. So, for the agent to better maximise the benefit in a changing environment, we add a ratio $mt = \text{BudgetLeft}/\text{TimeLeft}$ to dynamically adjust the intensity of exploration, making it reflect the rate at which the budget is spent. In this way, if there is an ample budget left and the episode is almost finished, then the agent can boldly explore all opportunities. Conversely, if the budget is consumed too fast, we can reduce ϵ to throttle the rate of exploration, so that the agent makes greedy choices according to the previously obtained information. However, it is not practical to use this ratio directly as a value of ϵ, because this ratio obeys a right-skewed distribution, is unevenly distributed, and most of the values are outside the $[0, 1]$ interval. Therefore, we apply a log transformation and min-max normalisation of the ratio so that the parameter is more uniformly distributed within the interval $[0, 1]$, denoted as mt'. Figure 2 shows the comparison of this ratio before and after transformation.

Intuitively, the strategy for dealing with the exploration and exploitation trade-off is as follows:

$$a_t = \begin{cases} \text{argmax}_a\, q(s_t,\, a) & \text{with probability } 1 - mt', \\ \text{random } a & \text{with probability } mt'. \end{cases}$$

3 Results and Discussion

This section reports on the simulated interactions results[3]. We run 1 000 episodes for each algorithm to train our agent. There are 500 time steps, i.e. 500 bidding interactions, per episode, and each episode starts with a total budget of $120.

[3]The experiment code is available at https://github.com/JingWen17/Budget-Constrained_RTB_Optimisation

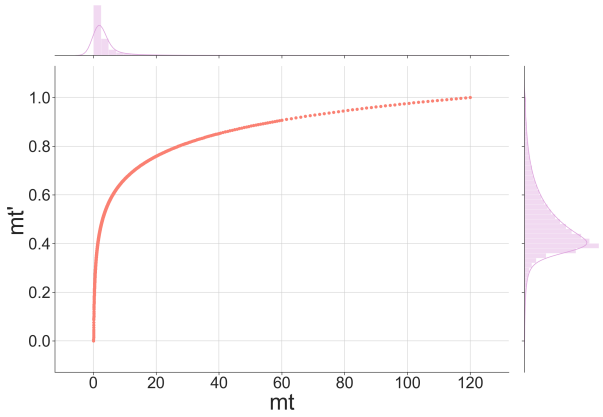


Figure 2: The scatter plot within the grid visualises the correlation between the values of mt before and after log transformation and min-max normalisation (mt'). The marginal plots represent their respective distributions. The distribution of mt at the top has a noticeable left skew, while the distribution of mt' is closer to a normal distribution.

Figure 3 shows the cumulative rewards obtained by the different algorithms. We begin by noting that the returns increase as more episodes are trained, which indicates that our agent is constantly improving its ability to recognise promising impressions. Secondly, the TD methods (Q-Learning and SARSA) perform better than the MC methods. This is likely because the TD approaches update their policies much faster, whereas the MC approaches need to wait until the end of each episode and then calculate the value of each encounter by reviewing the interaction trajectory. The most obvious example is that in a few beginning episodes, the agent is likely to be in a situation where nothing is known about most states, because it has not had enough opportunities to interact with the environment. At this point, the agent makes decisions primarily through random selection. If it happens to receive positive feedback in an interaction, the MC agent can only update its policy once the entire episode is finished. If this feedback points it in the wrong direction, then the MC agent may have to go through many more episodes to make up for the mistake. This is not the case with the TD approach, where the TD agent does not have to wait until the end of an episode before calibrating its knowledge of the environment, but instead adjusts its policy at every time step of the interaction.

Next, we compare our proposed algorithm with baselines in a dynamic environment, where the baseline Q-Learning and Sarsa algorithms consistently choose actions that yield the highest reward based on the estimated $q\langle s_t, a_t\rangle$, focusing solely on exploitation. In this experiment, the reward associated with a specific topic varies over time, and reaches its highest values towards the end of an episode, when the agent's budget might be running low. This reflects real-world fluctuations linked to, for example, time of day, season, and global events. The results are reported in Figure 4.

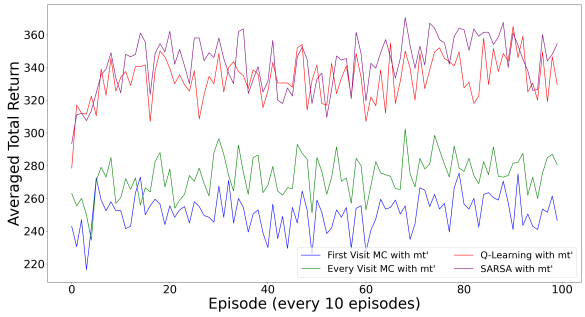
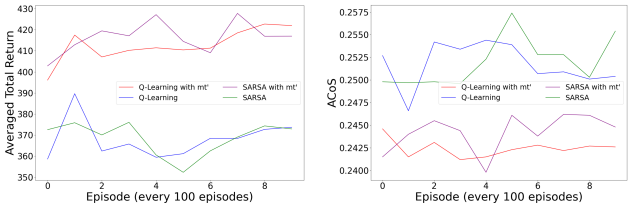


Figure 3: The total returns obtained using different algorithms with a changing epsilon (mt'). The x-axis and y-axis represent the count and average total return, respectively, per 10 episodes. The initial epsilon for all the algorithms is 0.1 and the learning rate for Q-Learning and SARSA is 0.25.



(a) (b)

Figure 4: Comparisons between the performances of Q-learning and Sarsa, both with and without the changing epsilon (mt') in a dynamic environment. (a): The average total return per 100 episodes. (b): The ACoS calculated based on the given formulation in Section 2.2.

By incorporating our proposed dynamic epsilon, both Q-Learning and Sarsa achieve higher total returns compared to the baseline algorithms. The dynamic epsilon stimulates exploration while the learner has a sufficient budget left to leverage new findings. In contrast, the baseline algorithms always take the currently optimal action. In addition, Q-Learning and Sarsa with mt' prove to be more effective and sustainable than the baselines. Figure 4(b) shows that four algorithms all achieve a good ACoS of around 24% to 26%. However, our proposed algorithms stand out by having a lower ACoS, which is valuable when budgets are limited.

In summary, our findings indicate that compared to MC methods, Q-Learning and SARSA are better able to adapt their strategies through interactive learning. In addition, our proposed dynamic epsilon method outperforms the baseline algorithms by more efficiently and sustainably learning in a dynamic environment. In the specific problem we simulated, which is characterised by a relatively small state space, limited unexpected events, and constrained time and budget, SARSA exhibited greater stability overall.

4 Conclusion

We model contextual RTB as a dynamic interaction process under budget constraints, define a suitable set of parameters to reflect real-world applications of contextual advertising, propose a dynamic epsilon-greedy algorithm to maximise long-term rewards sustainably, and compare the performance of different model-free reinforcement learning methods. Through experiments on real-world data, we can find that Q-Learning and SARSA outperform MC-based approaches to the problem. In addition, our proposed dynamic epsilon approach performs better in a changing environment. Future work includes the exploration of more advanced contextual features, for example, the semantic congruence between webpages and ads. This may involve further studies assessing the state-of-the-art text retrieval techniques and similarity measurement methods. Furthermore, we will also extend our bidding experiment to include several competing bidders.

References

[1] Preston Bukaty. 2019. *The California Consumer Privacy Act (CCPA): An implementation guide*. IT Governance Publishing. http://www.jstor.org/stable/j.ctvjghvnn

[2] Han Cai, Kan Ren, Weinan Zhang, Kleanthis Malialis, Jun Wang, Yong Yu, and Defeng Guo. 2017. Real-Time Bidding by Reinforcement Learning in Display Advertising. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining* (Cambridge, United Kingdom) *(WSDM '17)*. Association for Computing Machinery, New York, NY, USA, 661–670. https://doi.org/10.1145/3018661.3018702

[3] Graham Greenleaf. 2021. *Global Data Privacy Laws 2021: Uncertain Paths for International Standards*. Technical Report. 169 Privacy Laws & Business International Report 23-27. https://ssrn.com/abstract=3836408

[4] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: a factorization-machine based neural network for CTR prediction. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence* (Melbourne, Australia) *(IJCAI'17)*. AAAI Press, 1725–1731. https://doi.org/10.5555/3172077.3172127

[5] Tongwen Huang, Zhiqi Zhang, and Junlin Zhang. 2019. FiBiNET: Combining Feature Importance and Bilinear Feature Interaction for Click-through Rate Prediction. In *Proceedings of the 13th ACM Conference on Recommender Systems* (Copenhagen, Denmark) *(RecSys '19)*. Association for Computing Machinery, New York, NY, USA, 169–177. https://doi.org/10.1145/3298689.3347043

[6] Eugene Ie, Chih-wei Hsu, Martin Mladenov, Vihan Jain, Sanmit Narvekar, Jing Wang, Rui Wu, and Craig Boutilier. 2019. RecSim: A Configurable Simulation Platform for Recommender Systems. https://doi.org/10.48550/ARXIV.1909.04847

[7] Claudia Perlich, Brian Dalessandro, Rod Hook, Ori Stitelman, Troy Raeder, and Foster Provost. 2012. Bid Optimizing and Inventory Scoring in Targeted Online Advertising. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Beijing, China) *(KDD '12)*. Association for Computing Machinery, New York, NY, USA, 804–812. https://doi.org/10.1145/2339530.2339655

[8] Richard S. Sutton and Andrew G. Barto. 2005. Reinforcement Learning: An Introduction. *IEEE Transactions on Neural Networks* 16, 1 (2005), 285–286. https://doi.org/10.1109/TNN.2004.842673

[9] Michael Veale and Frederik Zuiderveen Borgesius. 2022. Adtech and Real-Time Bidding under European Data Protection Law. *German Law Journal* 23, 2 (2022), 226–256. https://doi.org/10.1017/glj.2022.18

[10] Di Wu, Xiujun Chen, Xun Yang, Hao Wang, Qing Tan, Xiaoxun Zhang, Jian Xu, and Kun Gai. 2018. Budget Constrained Bidding by Model-free Reinforcement Learning in Display Advertising *(CIKM '18)*. Association for Computing Machinery, New York, NY, USA, 1443–1451. https://doi.org/10.1145/3269206.3271748

[11] Weinan Zhang, Shuai Yuan, and Jun Wang. 2014. Optimal Real-Time Bidding for Display Advertising. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (New York, New York, USA) *(KDD '14)*. Association for Computing Machinery, New York, NY, USA, 1077–1086. https://doi.org/10.1145/2623330.2623633