Know Your Enemies and Know Yourself in the Real-Time Bidding Function Optimisation

Manxing Du*
University of Luxembourg
manxing.du@uni.lu

Alexander I. Cowen-Rivers
MediaGamma Ltd.
mc_rivers@icloud.com

Ying Wen
University College London
ying.wen@cs.ucl.ac.uk

Phu Sakulwongtana
University College London
zcabpsa@ucl.ac.uk

Jun Wang
University College London
jun.wang@cs.ucl.ac.uk

Mats Brorsson
University of Luxembourg
mats.brorsson@uni.lu

Radu State
University of Luxembourg
radu.state@uni.lu

***Abstract*—Real-time bidding (RTB) is a popular method to sell online ad space inventory using real-time auctions to determine which advertiser gets to make the ad impression. Advertisers can take user information into account when making their bids and get more control over the process.**

The goal of an optimal bidding function is to maximise the overall effectiveness of the ad campaigns defined by the advertisers under a certain budget constraint. A straightforward solution would be to model the bidding function in an explicit form. However, such functional solutions lack generality in practice and are insensitive to the stochastic behaviour of other bidders in the environment.

In this paper, we propose to formulate the online auctions into a general mean field multi-agent framework, in which the agents compete with each other and each agent's best response strategy depends on its opponents' actions. We firstly introduce a novel Deep Attentive Survival Analysis (DASA) model to estimate the opponent's action distribution on the ad impression level which outperforms state-of-the-art survival analysis. Furthermore, we introduce the DASA model as the opponent model into the Mean Field Deep Deterministic Policy Gradients (DDPG) algorithm for each agent to learn the optimal bidding strategy and converge to the mean field equilibrium.

The experiments have shown that with the inference of the market, the market converges to the equilibrium faster while playing against both fixed strategy agents and dynamic learning agents.

***Index Terms*—Real-time bidding, Multi-agent, Deep Reinforcement Learning**

I. Introduction

Real-time bidding (RTB) is a common online advertisement (*ad* for short in the rest of the paper) inventory trading mechanism in which each ad display is sold through *real-time auctions*. It allows the advertisers to target potential users at the level of individual ad impressions. Upon each user's visit, each ad slot on the publisher's site (or app) are sold through auctions. Most market-places use a so-called *second price auction* [1]. In such auctions, the bidder with the highest bid price wins the opportunity to show its ad, which is also called an *ad impression* and the winner observes and pays the second highest price, known as the *market price* or the winning price.

*Work conducted during a research visit at MediaGamma.

The market price is only really known by the highest bidder, the winner. The other bidders, who lost the auction, only know that the market price could be equal to or higher than their bids. Therefore, the market price can be considered to be right-censored. In addition, the bidding environment is highly dynamic. In each auction, an unknown number of bidders will participate and the set of bidders varies over different auctions. To compete in such second price auctions, in theory, bidders would benefit from bidding their estimation of each impression's true value as the bid price [1]. In this way, the environment converges to the Nash equilibrium in which no bidder can benefit more by unilaterally changing its strategy.

However, in practice, the market may not always maintain the ideal equilibrium due to various reasons. For instance, since the bidders are usually constrained by a certain budget, to avoid running out of money quickly without observing more valuable impressions, the optimal bid price usually deviates from its true value. In addition, the number of participants in each auction is unknown and from each bidder's perspective, it may compete with different opponents at every step during its lifetime. To obtain an optimal bidding strategy in such a stochastic game with a large number of unknown participants is the major challenge in RTB.

The dynamics in the RTB market mostly come from the heterogeneous behaviour of each bidder. It is important for an intelligent bidding agent to try to infer its opponents' strategy while optimising its own strategy. With the number of opponents increasing, modeling every opponent's action becomes implausible and computationally expensive. To analyze such highly dynamic games with large number of participants, recent research work [2, 3, 4] has been focused on using Mean Field Equilibrium (MFE) to approximate the Nash Equilibrium. Instead of estimating the interactive strategies between each agent pair, it simplifies for one player to react to the estimated average action from all other players.

We have designed, implemented and evaluated a bidding algorithm which is *opponent-aware* but still requires no prior assumptions on the opponents' bidding distribution. To the best of our knowledge, this is the first known such algorithm.

We firstly address the prediction of partially observable

opponent actions and adopt a Deep Attentive Survival Analysis (DASA) model which greatly outperforms the state-of-the-art survival models. Furthermore, our solution integrates the opponent model into the policy learning framework for actor-critic based bidding agents. We take the second highest price as the aggregated action of all the other bidders, which enables each bidder to optimise its policy together with modeling the uncertainty of the market. Our experiments shows that we reach equilibrium under different budget constraints and improved convergence in the multi-agent environment. Furthermore, we demonstrate the performance improvements of the optimal MFE strategy from a single agent's perspective.

II. Related Work

Bid Optimisation. Bid optimisation is one of the key components in the decision making process in RTB. It aims to optimise the bid price on the impression level which maximises the potential profits under a certain budget constraint. Many research works have formulated it as a functional optimisation problem [5, 6]

Zhang et al. [6] models the optimal bidding function in a concave form which nonlinearly correlates the bid price with the predicted click through rate. However, the functional based methods have strong assumptions of the model form and fail to incorporate the dynamics in the bidding environment and the bidder's budget spending status into the model.

To address the above shortcoming, there has been some focus on Reinforcement Learning (RL) based methods [7, 8, 9], which directly formulate the bidding strategy as a sequential decision making process without a predefined function form. These studies mainly address the bidding optimisation problem for a single agent.

In [7], Cai et al. proposed a model-based Markov Decision Process (MDP) to formulate the interaction between one bidding agent and a *stationary* environment. It assumes that all other bidders and the users are part of the same environment. In this work, it is necessary to calculate the state transition probability to derive the optimal policy. This calculation is computationally expensive when the dataset is large. To avoid the computational cost, other researchers have adopted model-free reinforcement learning methods and also extended the single agent learning to multi-agent learning [10, 11].

In multi-agent environments, from each agent's view, when its opponents adapt their behaviour, the changes affect each other and the environment becomes non-stationary. To address the non-stationary environment becomes the main challenge in the multi-agent learning. It is essential for each agent to account for how other agents react and take the joint behaviour to learn its own strategy. Such scenarios are called stochastic games, which require equilibrium-solving approaches for the policy learning. These approaches can be categorized by how the non-stationary behaviour are handled [12]. In most of these studies, the learning agent assumes its opponents are acting in a specific way, either using a fixed strategy to minimize each other's reward [13] or using a mixed strategy drawn from a set of known strategies [14]. In case the strategy is not explicitly defined, Nash-Q learning is introduced with the assumption that all the agents are adapting their strategies to converge to the Nash-Equilibrium [15], which is called the NE strategy. In practice, the Mean Field Equilibrium (MFE) is used to approximate the Nash Equilibrium when the number of agents is large [2].

In RTB, the existence of MFE under budget constraints has been theoretically proved in [3, 16]. Both studies showed that to reach the MFE, each agent takes the known fixed market price, budget, and the observed reward distribution (e.g. the estimated click through rate) to estimate the value function. In [10], Jin et al. aggregated bidders into clusters and applied the Deep Deterministic Policy Gradients (DDPG) algorithm on the cluster level (as one agent) to simulate the multi-agent environment. It demonstrates the profit gain per bidding cluster under the competing or the collaborating settings. The similar assumption applies that each agent knows each other's state, action, and reward. In practice, the states (e.g. the budget, the current obtained reward) of the other bidders are usually unknown. The action of others are partially observable through the market price only to the winner of each auction.

In our work, we provide a multi-agent bidding strategy by extending the work in [10] with the generalization of playing with partially observable opponents. Given that the number of bidders in the bidding market is large and their strategies remain unknown, inspired by the MFE theory, we aggregate the set of bidders who bid the market price in each auction as an virtual agent and introduce a market price prediction model as the Opponent Model (OM). From each agent's view, we assume all the other agents adapting their strategies towards the mean field equilibrium without any irrational drastic changes.

Bid Landscape Forecasting. Market price prediction has been widely studied in RTB [17, 18, 19]. One key challenge is that in the second price auctions, the second highest price, a.k.a the market price, is only shown to the winner and remains unknown to the others. Thus, the market price is right-censored. To address the data censorship, a survival model is commonly applied to estimate the time until the occurrence of a particular event, for instance, the survival time of patients in the medical domain [20]. In RTB, the market price estimation has been commonly addressed by adopting a non-parametric Kaplan-Meier estimator for the entire dataset [18]. However, one aggregated distribution for all the bid requests fails to capture the divergence in the feature space. In [17], the authors proposed to adopt the recurrent network to model the sequential pattern in the feature space of the individual user and estimate the market price distribution for each bid request. However, the features may not only be limited to the sequential dependencies. The transformer model [21] is the first model relying entirely on self-attention to compute a representations of its input without using convolutions or sequence aligned recurrent neural networks [22]. It has fueled much of the latest development, such as pre-trained contextualized word embeddings [23, 24, 25]. In this work, we adopt the transformer model to generalize the sequential dependency in the feature

space by using the attention mechanism.

III. Problem Formulation

In RTB, when a user visits a website, an online auction is held by an ad exchange to sell the ad slots available on the web page. A bid request containing the context of the ad slot, the user's and the publisher's profile will be distributed to the Demand Side Platforms (DSPs). On behalf of the advertisers, each DSP selects the best match of the ad from its inventory and sends a bid price back to the exchange. The DSP with the highest bid price wins the auction and the selected ad will be shown to the user as an ad impression. The winner of the auction pays the second highest price among the submitted bids, which is also called the market price. In the bidding environment, each bidder is unaware of the status of other bidders and can decide whether to participate the auction or not. Consequently, in each auction, the set of competing bidders may vary and remain unknown to each other. The only information each bidder can know about others is through winning the auction and observing the market price. The market price remain censored to all the bidders except the auction winner. Each bidder usually starts with a limited budget and the goal is to bid optimally which maximises its profit.

In this section, we formulate the sequential second price auctions as a *n*-player stochastic game represented by a tuple $\Gamma =< \mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{P}, \gamma >$. Here, the players are the bidders who share the same bidding environment. The environment dynamics come from the simultaneous actions, a.k.a the bid prices sent by the bidders. We refer the bidders as the agents interchangeably in the rest of the paper. At each time t, the state of all agents is denoted as $\boldsymbol{s}_t = (s_t^1 \dots s_t^n) \in \mathcal{S}$. Correspondingly, the joint action of all agents at time t is represented as $\boldsymbol{a}_t = (a_t^1 \dots a_t^n) \in \mathcal{A}$. The policy of an agent is defined as: $\pi^i : \mathcal{S}^i \mapsto \mathcal{A}^i$. After taking an action a_t^i, agent i transits to the next state: $s_{t+1}^i \sim \mathcal{P}_t^i(s_t^i, \boldsymbol{a}_t^i)$ and receives a reward $r_t^i \sim \mathcal{R}_t^i(s_t^i, \boldsymbol{a}_t)$. We note that the transition probability and the reward are determined by the joint action $\boldsymbol{a}$. Since all the agents set their price simultaneously, from each agent's point of view, only the winner pays the market price and the others keep their budget to the next auction. Correspondingly, only the winner has the chance to get the user behaviour dependent reward, e.g. the click through rate (CTR) and the others get zero benefits.

Unlike in other *n*-player games, where the environment state is usually considered in the equations above, in RTB, each agent only observes its own state s_t^i at each step. The agents are coupled only through their actions.

The value function of a certain policy π is defined as:

$$v_\pi^i(s) = \sum_{t=0}^{\infty} \gamma^t \, \mathbb{E}_{\pi,\mathcal{P}}[r^i(s_t, \boldsymbol{a}_t) | s_0 = s, \pi], \tag{1}$$

where $\gamma \in [0, 1)$ denotes the reward discount factor over time.

As stated in [2], a common assumption of such a *n*-player game is that each agent is unaware of the game dynamic or the reward function, but it observes the previous actions and the immediate reward of other agents. For a single agent, the Q-function is extended by taking the joint actions of all agents $\boldsymbol{a} \triangleq [a^1, \dots, a^n]$ as the formulation below:

$$Q_{\boldsymbol{\pi}}^i = r^i(s, \boldsymbol{a}) + \gamma \, \mathbb{E}_{s' \sim \mathcal{P}}[\mathbb{E}_{a \sim \boldsymbol{\pi}}[Q_{\boldsymbol{\pi}}^i(s', \boldsymbol{a})]]. \tag{2}$$

The objective of each agent in an *n*-player game is to derive an optimal policy π_* which maximises its value function as shown in Eq.1. However, since each player has its own reward function, which is dependent on the other players. Hence, it is not a clear concept defining an optimal policy as in single-agent problems to only maximise the state value given in Eq.1. It is may be ineffective due to players have different reward and one agent might hamper the objective of another, the bidding environment becomes non-stationary as well.

To address this problem, one common solution is equilibrium-based approaches. We start from one agent's view, the stochastic formulation of RTB can be simplified as a two players game, where the other player is the one with the second highest price in the market. The winning price can be modeled as the joint action from all the other opponents and is partially observable. Then, we adopt equilibrium-based policy is also called the Nash equilibrium (NE) policy [26] which satisfies

$$v^i(s, \boldsymbol{\pi}_*) \geqslant v^i(s; \pi^i, \pi_*^{-i}). \tag{3}$$

where $\boldsymbol{\pi}_*$ is the NE optimal joint policy of all the agents and no agent can further improve its value function while other agents keep their policies unchanged. Here π_*^{-i} denotes the optimal joint policy of all the other agents except agent i: $\boldsymbol{\pi}_*^{-i} \triangleq [\pi_*^1, \dots, \pi_*^{i-1}, \pi_*^{i+1} \dots, \pi_*^N]$.

When the number of agents $n \to \infty$, the classic multi-player game becomes intractable, thus in [27], the authors proposed the Mean Field Game (MFG) to model the large number *n*-player game. The conventional MFG assumes the agents have complete information of the actions and the rewards of other agents [2]. On the contrary, in RTB, the actions of other agents are not observable unless the agent wins the auction and the highest bid from the other bidders is revealed. To generalize the conventional MFG to the MFG with incomplete information, in this study, we propose an opponent model to infer the unobservable actions of other players. From one agent's perspective, at each time t,

$$s_{t+1}^i \sim \mathcal{P}(\cdot | s_t^i, \boldsymbol{a}_t) = \mathcal{P}(\cdot | s_t^i, a_t^i, \boldsymbol{a}_t^{-i}). \tag{4}$$

where $\boldsymbol{a}_t^{-i}$ denotes the actions taken by the other agents other than agent i. If here all the other agents follow their optimal policies, so that $\boldsymbol{a}^{-i} = \boldsymbol{a}_*^{-i}$. It suggests the bid distribution of other agents is fixed. Therefore, Eq.3 can be written as

$$v^i(s, \pi_*^i, \boldsymbol{a}_*^{-i}) \geqslant v^i(s; \pi^i, \boldsymbol{a}_*^{-i}). \tag{5}$$

To learn an equilibrium joint policy, updates of Q-values rely on the computation of an equilibrium metrics, Nash-Q, defined in [15]:

$$\boldsymbol{Q}^{\text{Nash}}(s, \boldsymbol{a}) = \mathbb{E}_{s' \sim \mathcal{P}}[\boldsymbol{r}(s, \boldsymbol{a}) + \gamma v^{\text{Nash}}(s')]. \tag{6}$$

The Q-function in Eq.6 is approximated by neural networks and parameterized with the weights ω. As discussed above, we

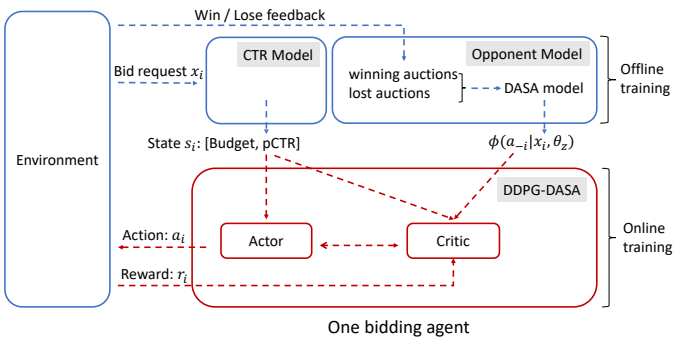

Fig. 1: The architecture of the DDPG-DASA model.

replace the joint actions of other bidders $\boldsymbol{a}_t^{-i}$ by a virtual and aggregated competitor whose bids are always the market price. Thus, the Q-function can be expressed as

$$Q^i(s, \boldsymbol{a}) = \int_{a^{-i}} Q(s, a^i, a^{-i})\phi(a^{-i})\ da^{-i} \tag{7}$$

In the ideal MFE scenario, it supposes that all the agents take a fixed and steady bid distribution and their own belief of the bid valuation as the prior knowledge to optimise their strategy[16]. The policy that each agent follows is stationary. In practice, the bid valuation is estimated by the CTR prediction model and the opponent bid distribution can be estimated as the market price model. Thus, in this work, we adopted two pre-trained models into the framework to fulfill the above assumption.

Fig. 1 depicts the architecture of the components used in this work. The CTR model takes the feature vector **x** in the historical bid requests as input and binary labels 1 and 0 indicating an ad click or no click respectively. The predicted Click Through Rate (pCTR) is later used to construct the agent state s and the reward r in the updated DDPG model with the opponent model DASA (DDPG-DASA). In the following sections, the opponent model (DASA) and the bidding model DDPG-DASA are described in details.

A. Opponent Modeling

In this section, we focus on modeling the opponents actions, a.k.a the market price distribution. The opponent model is defined as the market price distribution at an impression level. We use a^{-i} to represent the action taken by the opponents, a.k.a the highest price from all the other participants in the auction. In this section, we denote $a_{-i} = z$, where z is the market price. The probability density function (P.D.F) of z is $p_z(z)$.

$$p_z(z) = P(a^{-i} = z|\boldsymbol{x}_i, \theta) = h_z \prod_{j<z}(1 - h_j). \tag{8}$$

As is shown in Eq. 8, the P.D.F of the market price can be calculated from the instant hazard function h_j which indicates the probability of the instant occurrence of the event at time j conditioned on the event has not happened prior to time j. In the RTB setting, $\prod_{j<z}(1 - h_j)$ represents the losing probability of bidding less than the market price and h_z shows the probability of observing the market price z.

We take the features in the bid request $\boldsymbol{x}_i$ as the input and predict the hazard function h over the discretized bid price space at each impression level. The p_z can be easily derived from Eq. 8. We use b_{max} to represent the upper bound of the bid price. For the uncensored data, the true label is an one-hot encoded vector of size b_{max} with the element indexed by the market price as 1.

We followed the loss functions in [17], for the uncensored data, the loss of the observed market price is defined as:

$$L_z = -\sum_{\boldsymbol{x}_i, a_{-i} \in \mathcal{D}_{\text{uncensored}}} [log h_z + \sum_{j<z} log(1 - h_j)].$$

For the censored data, it is certain to still lose the auction by bidding lower than the current price. The corresponding loss is defined as:

$$L_{\text{censored}} = -\sum_{\boldsymbol{x}_i, a_i \in \mathcal{D}_{\text{censored}}} \sum_{j<a_i} log(1 - h_j).$$

In addition, for the winning auctions, by bidding at any price higher than the observed market price, it is guaranteed to win the auction. Such information can be shared with the censored data. The loss function is defined as followed:

$$L_{\text{uncensored}} = -\sum_{\boldsymbol{x}_i, a_i \in \mathcal{D}_{\text{uncensored}}} log[1 - \prod_{j<a_i}(1 - h_j)].$$

The total loss of the model takes the combination of the above losses as below where α balances the loss values.

$$L_{\text{total}} = \alpha L_z + (1 - \alpha)(L_{\text{censored}} + L_{\text{uncensored}}).$$

B. Bidding Model

Under our repeated second-price auctions setting, in every auction, all the agents are facing the same bid request. The agents bid for the same ad campaign upon different requests with unknown number of opponents at each auction. The RL agent adopts the framework of Deep Deterministic Policy Gradient (DDPG) [28] method to learn the policy in a continuous space.

State. For the DDPG agent, we take the budget left in an episode B_i and the $pCTR$ as the state $s =< B_i, pCTR >$

Action. Following the settings in [10], the action a_i is set to be a scaler which controls the bid price and is bounded to be in the range of $[0, 1]$. The final bid price is calculated by $b_f = \min(b_{max} \times a_i, B_i)$, where b_{max} is the upper bound of the bid price. The market price or the aggregated actions from the opponents are denoted as a_{-i}.

Reward. The reward is usually the Key Performance Indicator (KPI) defined by the advertisers, for instance, a click, a purchase or the profits. But such reward signal is usually too sparse for the agent to learn. Therefore, in this study, we assign the $pCTR$ as the *Shaped Reward* for all the winning auctions, even without the real click [9]. For the losing auctions, since no price is paid, the reward remains as zero. For agent i, the actor network takes state s, which consists of the predicted CTR and the budget left in the current episode, and parameterized with

θ_π for a deep neural network which provides an action to take in the range of $[0, 1]$.

Action function

$$a_i = \pi_i(s_i, \theta_\pi) = \pi_i([b_i, pCTR_i], \theta_\pi). \tag{9}$$

In the vanilla version of DDPG algorithm, the critic function $Q(s_i, a_i)$ takes the state and action pair from a single agent. In our model, the Q-function is approximated by the mean field theory by integrating the opponent's action distribution. As is shown in Eq. 10, $\phi(a_{-i}|\boldsymbol{x}_i, \theta_z)$ is the market distribution obtained from the opponent model. The action a_{-i} is not directly observed from the environment, since the result of the auction can only be see after placing a bid price. The market distribution provides the agent's belief of the opponents actions. The indicator function allows the agent to account for the Q value only in the case of bidding higher than the market price. Since when the action a_i is lower than a_{-i}, the agent cannot win such auctions, thus, the Q value should be zero.

Critic function

$$Q(s, \boldsymbol{a}) = \int_{a_{-i}} Q(s, a_i, a_{-i})\phi(a_{-i}|\boldsymbol{x}_i, \theta_z)\mathbb{1}[a_{-i} < a_i]\ da_{-i}. \tag{10}$$

The pseudo code of the DDPG-DASA algorithm is shown in Algorithm 1.

Algorithm 1: DDPG-DASA.

Initialize actor network $\pi(s, \theta_\pi) = a_i$ and critic network $Q(s, \mathbf{a}|\omega)$ with weights θ_π, ω
Initialize target network π' and Q' with $\theta'_\pi \leftarrow \theta_\pi$ and $\omega' \leftarrow \omega$
Initialize replay memory with size K;
for *episode = 1 to E* **do**
  receive state s_0 and sample $a_0 \sim \pi(s_0, \theta_\pi)$;
  Initialize a noise generator $\mathcal{N}$ for action exploration
  while s_i *not terminate* **do**
    Select an action $a_i = \pi(s_i, \theta_\pi) + \mathcal{N}_i$ and execute ;
    Observe r_i, s_{i+1};
    Store (s_i, a_i, r_i, s_{i+1}) in the replay memory;
    if $t \equiv 0$ *mod* K **then**
      sample a minibatch $\mathcal{M}$ from the replay memory
      $y_j = r_j + \gamma Q'(s_{j+1}, \boldsymbol{a'_{j+1}}, |\omega')$
      update critic by minimizing the loss $L = \frac{1}{\mathcal{M}}\sum_j (y_j - Q(s_j, \boldsymbol{a_j}|\omega))^2$;
      update actor $\theta \leftarrow \theta + \frac{1}{\mathcal{M}}\sum_j \nabla_a Q(s_j, \boldsymbol{a_j}|\omega)|_{s=s_j, a=\pi(s_j)} \nabla_{\theta_\pi}\pi(s_j|\theta_\pi)|_{s_j}$;
      update target network:
      $\theta' \leftarrow \tau\theta_\pi + (1-\tau)\theta'_\pi$;
      $\omega' \leftarrow \tau\omega + (1-\tau)\omega'$
  end
end

C. Multi-Agent Mean Field Approximation

The mean field equilibrium in RTB requires a consistency check of the bid distribution [16]. Let ϕ be a bid distribution and π_i denote a stationary policy for an agent facing bidding decision. The mean field equilibrium is achieved if it satisfies the following definition:

Definition 1: The repeated second-price auction Mean Field games admit at least one *mean field equilibrium* (MFE)[16], with strategy π, if:

1) $\pi(\cdot|\phi)$ is an optimal strategy given ϕ.
2) ϕ is the steady state bid distribution given π.

1) Single-agent Steady Market Distribution: We start from the simplest scenario: a single agent bids against the steady market price distribution. In this setting, we assume the linear bidders have fixed strategies which means they do not update their strategies upon other bidders' actions. In addition, given the dynamic attributes of the bidders, from agent i's point of view, the bids from its opponents are identically and independently distributed. As we discussed in I, in practice, the opponent sets in every auction changes over time. Here we assume the departure and the arrival rate of bidders remains steady, which guarantees the stationary of the bid price distribution of the opponents. Even that the opponent bids are partially observable, this allows us to approximate a fixed opponent model and use it in the mean field model.

2) Multi-agent Dynamical Market Distribution: From the above single agent scenario, here we extend to discuss the mean field equilibrium. Instead of considering only one agent, we focus on the multi-agent environment, where all agents assume to share one steady bid distribution ϕ. Taken ϕ as the prior knowledge, each agent optimises their bidding strategy which in turn induces dynamics in the overall bid distribution.

In this game, we assume that the number of competing agents is large. For each auction, a finite number of agents is randomly selected (through Gaussian noise randomly selected a competing agent, the agent with the largest noise added one is effectively selected). Each agent has a random life-time, which is exponentially distributed with unit mean. It optimises the utility over its lifetime. The unit mean is effectively the fact that each agent starts with the same budget, however the varying lifetime depends on the pCTR values estimated and also the exponentially distributed additive noise. At either the end of the episode or when the budget has been exhausted, agents are replaced by new ones whose initial budget, valuation distribution and income is sampled. In most experiments, instead of randomly sampling budget we initialized this to the same value, and noticed no difference in convergence guarantees. Due to a learning rate decay, eventually the DDPG agent will converge to a stationary agent (learning rate ≈ 0), thus the normal theorem by [16] holds.

In the MFE, each bidder are facing i.i.d highest opponent bids and has no incentive to change her bidding strategy. However, it is important to note that before the equilibrium is reached, the bid distribution would change as the market

evolves. Thus it is important for the agents to infer the bid distribution over time.

IV. Experiments

In this section, we firstly demonstrate the significant improvement of the DASA model over the state-of-the-art market modeling methods. Then, we integrate the DASA model as the opponent model into the modified DDPG algorithm, we present the empirical study of the DDPG-DASA bidding algorithm in both single-agent and multi-agent scenarios on the large-scale real-world bidding dataset. We have published the implementation code the experiments[1].

A. Opponent Model

We first conduct experiments to compare the general behaviour of the DASA model with other survival analysis models. The DASA model consists of one transformer encoding layer. A transformer encoder layer has two sub-layers. The first is a multi-head attention mechanism, the second is a fully connected feed-forward network. Residual connections [29] are used around each of the two sub-layers, followed by layer normalization [30]. All sub-layers in the model produce outputs of dimension 512 in order to facilitate residual connections. We use a fixed learning rate of 0.001, state size of 128, batch size of 256 and 8 heads of multi-head attention.

The experiments are conducted on 3 datasets: Clinic[31], Music[32], and Bidding dataset. The statistics of the datasets can be found in [17]. We reproduced the results in [17] by using the publicly available code[2] and datasets[3] as the baseline results with * in Table I. The evaluation metric is the average negative log probability (ANLP) of the market price which corresponds to the true market price likelihood loss. It is defined as:

$$A = -\frac{1}{|\mathcal{D}_{test}|} \sum_{(\mathbf{x}_i, z_i) \in \mathcal{D}_{test}} \log p(z_i|\mathbf{x}_i) \tag{11}$$

where z_i is the market price from the simulation, x_i is the feature vector of a bid request. The lower the ANLP value gets, the better.

The result shows the DASA model significantly outperforms other methods across all three datasets. Thus, we select DASA model as our opponent model for the experiments in the following sections.

B. Bidding Experiments

1) Datasets and Experimental Setup: In this work, the bidding experiments are conducted over the public real-world dataset, iPinYou, one of the leading ad companies in China. The dataset records more than 15 million impressions in the original bid logs and the labels of click and purchase as the user feedback. The data are from 9 ad campaigns over a week in 2013 and the training and test data are separated by time [33]. In each bid request, it contains features of users (e.g. time stamp, segmentation label, and browser) and publishers (domain, ad slots, and URL). The statistics of the ad campaigns selected in the study are shown in Table II.

TABLE I: Performance comparison on and ANLP. DASA gets significantly improvement over strong baselines.

Models	ANLP		
	CLINIC	MUSIC	BIDDING
KM*	9.012	7.270	14.012
Lasso-Cox*	5.307	28.983	34.941
Gamma*	4.610	6.326	5.941
STM*	3.780	5.707	4.977
MTLSA*	17.759	25.121	9.979
DeepSurv*	5.345	29.002	35.405
DeepHit*	5.027	5.523	5.513
DRSA*	3.337	5.132	4.598
DASA (One Stack Transformer)	**2.786**	**4.912**	**3.465**

TABLE II: Dataset Statistics.

campaign ID	Impressions	Clicks	CTR	# Discrete Feature Values
2259 (Trainning Set)	835,556	280	0.034%	40347
2259 (Test Set)	417,197	131	0.031%	40347

We follow the data pre-processing and feature engineering procedure in [34]. Since in the iPinYou dataset, it records the original market price of the impressions, we initiate all the agents with the budget to be proportional to the total cost in the training data. In this way, it allows us to simulate the auctions offline. Given each bid request, each agent places a bid price and follows the second-price auction principles to decide the winner of the auctions and the click labels in the log are used to train the CTR model. We compare the bid price generated by the agents in our experiment, thus the original market price in the iPinYou bidding log is not included.

As is shown in Fig. 1, the CTR estimator is trained offline by adopting the widely used FTRL-logistic regression model [35]. In both single and multi-agent scenarios, we begin with running the bidding simulation over the training set and log the bid price of each agent and select the second highest price as the market price. The opponent model in Fig. 1 takes the simulated bid log and the features in the original bid requests as input to predict the impression level market distribution as described in Section III-A.

Once the CTR model and the opponent model are trained, we repeat the bidding simulation on both training and test sets. In this round, the DDPG agent learns the policy while having the prediction of the distribution of its opponent. We begin with setting one DDPG agent in the environment and keep the other bidders using simple and static bidding strategies, for instance, linear bidding function. In this setting, we demonstrate the advantage of the learning agent over the static agent without the learning process. Furthermore, we extend to the multi-agent scenario where all the agents learn their strategies with its estimated opponent model.

In this study, we consider the bidding process as an episodic task and each episode consists of $K = 1000$ auctions. Each episode has a fixed budget $B = \text{CPM}_{\text{train}} \times 10^{-3} \times K \times c_0$, where $\text{CPM}_{\text{train}}$ is the cost per mille impressions in the training

[1]https://github.com/manxing-du/Know-your-enemies.git

[2]https://github.com/rk2900/drsa.

[3]https://www.dropbox.com/s/q5x1q0rnqs7otqn/drsa-data.zip?dl=0.

data and c_0 is the budget constraint ratio: c_0= 0.125, 0.25, and 0.5.

2) Single DDPG agent with Steady Market Distribution: In this section, we assume that there is one learning agent running DDPG algorithm and competing against N bidders with fixed strategies, for example, a linear bidding function: $b_i = pCTR * \alpha_i$, where α_i is a fixed linear ratio. In practice, N is always unknown and in each auction, a random set of the N bidders is selected. In the second-price auction, the most important opponent is the bidder with the second highest price among all the bidders. In this study, we set up two linear bidders and one DDPG bidder. The bidders take the same *pCTR* from the CTR prediction model. By injecting Gaussian noises into the *pCTR*, we simulate the stochastic environment of random bidders with different *pCTR* as their states in each auction. Comparing the bid price generated by the three bidders, we log the market price and use it for training the DASA model offline as described in Sec.IV-A.

In the next round, we replay the bidding game again to train the same DDPG agent from scratch with the opponent model integrated, a.k.a the DDPG-DASA model. We run the same experiment with three random seeds and show the averaged results as follows. In Fig. 2, the number of clicks obtained by the three bidders are listed for one selected ad campaign, 2259. In general, the agent with the DDPG-DASA model obtains more clicks than the vanilla DDPG model when competing with the fixed linear bidders. In Fig. 3, it shows the number of impressions each agent obtains. Same as in Fig. 2, the rows represents three budget settings, where c_0= 0.125, 0.25, and 0.5. The left column shows the results from DDPG agent without the opponent model as the baseline while the right column shows the results of the agent with the DDPG-DASA model. The DDPG-DASA agent converges to the equilibrium faster than without the opponent model. The learning curve of the Q function shown in Fig. 4 confirms the faster convergence. The result suggests the benefits for the bidding companies who newly join the auctions.

We need to note that, the budget was set by referring to the original market price in the iPinyou bidding log. But the new market price generated by the agents are different and lower. At the beginning of their campaign lifetime, without any information of the market, the learning agent converges to a steady but sub-optimal strategy. However, if they infer the opponents model quickly and the opponents have fixed strategy, the DDPG-DASA model facilities the bidder to converge to a more dominant strategy which obtains more clicks in the market. If the other agents adopt learning process into their strategies which evolves the bid distribution, the challenge would be to show the asynchronous best response from all the agents and converge to the MFE which is shown in the next section.

3) Multi-agent game: In this section, the experiment is extended to have multiple learning agents in the same environment. Assume the number of agent $N = 3$, same as in the previous section, the three agents have the same budget setting for every 1000 auctions as one episode. As is shown

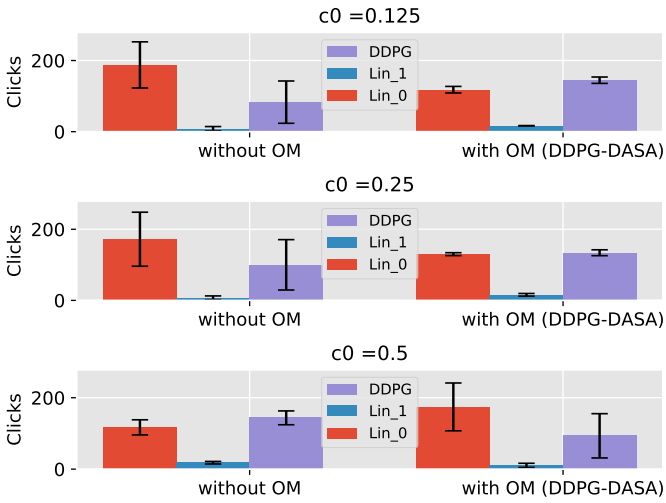

Fig. 2: The number of clicks won by every bidder of Campaign 2259, where Lin_* refers to the linear bidders.

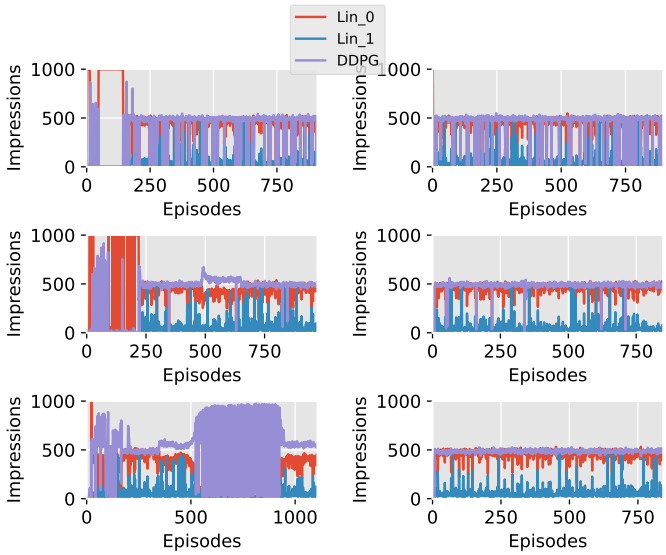

Fig. 3: The number of impressions won by every bidder of Campaign 2259.

in the first row in Fig. 5, the three agents start with bidding by only learning from its own reward without referring to other bidders' behaviour. After 200 episodes, the game converges to the equilibrium where the number of impressions won by each agent roughly evenly distributed. We continue the experiment by introducing a random market distribution as the opponent model for each agent. For each auction, the market distribution is sampled from a uniform distribution. The second row in Fig. 5 shows that the random opponent model increased the variance of the number of winning impressions for each agent and some agent may converge to a dominating strategy. To equipped the correct opponent model, we take the bidding log generated by the first game and trained a market model separately for each agent based on the set of impressions they

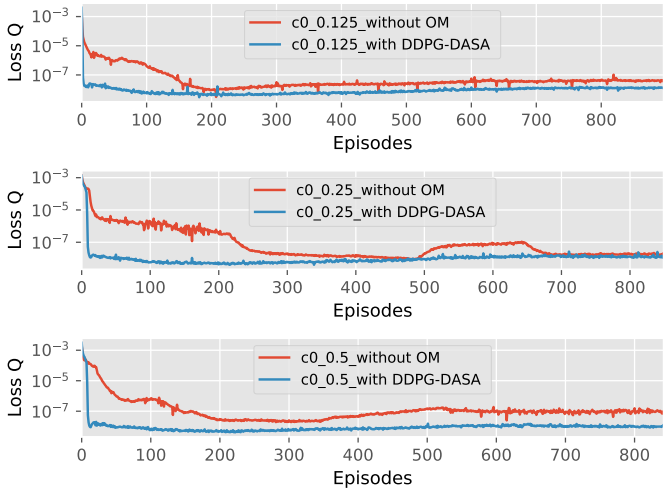

Fig. 4: Learning curves over Campaign 2259 under different budget settings.

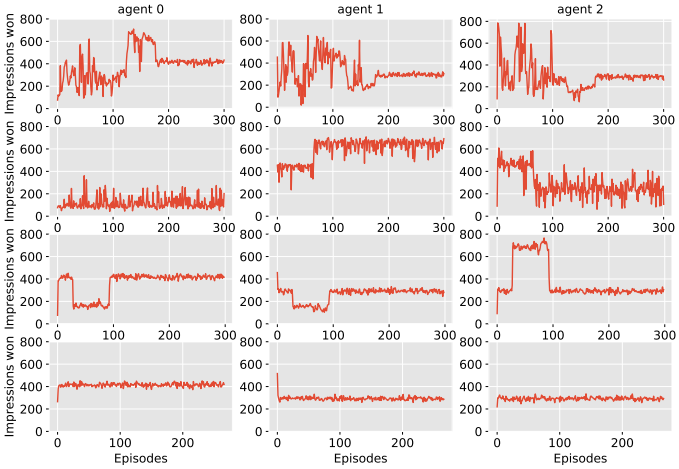

Fig. 5: Three DDPG agents bidding game. Row 1: DDPG without opponent model. Row 2: DDPG with random market model. Row 3: DDPG-DASA model on the training set Row 4: DDPG-DASA model on test set.

won. With the information of the market, we reset the game for the training set, as is shown in the third row in Fig. 5. The agents converge to the optimal strategies within 100 episodes which is 50% less than the results in the first row. We further test the model on the test set, which shows the model is generalized well and the equilibrium is reached.

Fig. 6 shows the number of clicks each agent won, which stays roughly the same. It demonstrates that since the reward function is not changed, once the equilibrium is reached, the multiple learning agents do not gain more profits. The benefits introduced by using the DDPG-DASA model is the fast convergence speed. In Fig. 7, it shows the DDPG-DASA model for each learning agent converges faster than the DDPG model.

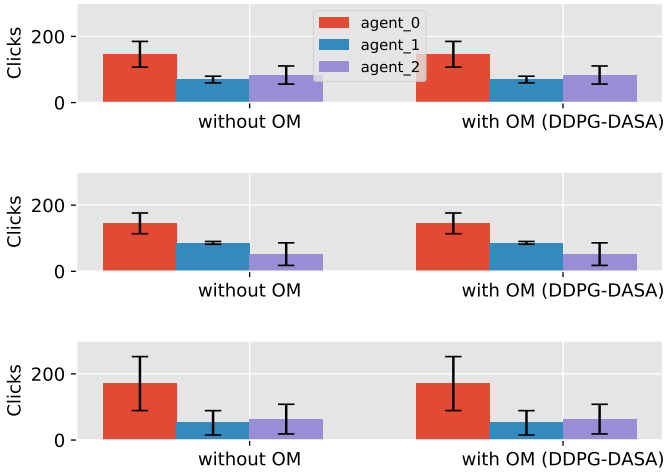

Fig. 6: The number of clicks each learnign agent gets in the multi-agent secenario.

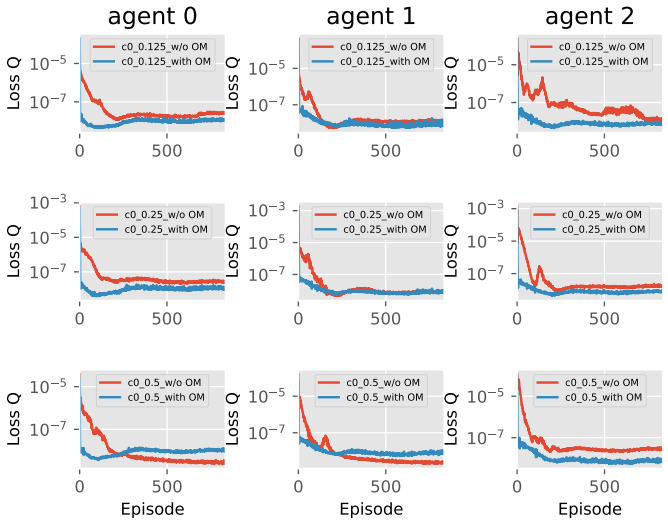

Fig. 7: Learning curves of DDPG-DASA model for each learning agent.

V. Conclusions

In this paper, we propose a general opponent aware bidding algorithm with no prior assumptions on the opponents bidding distribution. To the best of our knowledge, it is the first experimental implementation in the real-time bidding domain to infer the partially observable opponents in the policy learning process. We proposed a deep attentive survival model as the impression level opponent model. The multi-agent bidding simulations show the benefits of improved convergence rates for the DDPG model across all budgets with augmented with an opponent model. For the future work, instead of using pre-trained model, we will investigate the adaptive training for both the opponent model and the reward function in the multi-agent bidding game.

ACKNOWLEDGMENT

This research has been supported by the National Research Fund (FNR) of Luxembourg under the AFR PPP scheme. The experiments presented in this paper were carried out using the HPC facilities of the University of Luxembourg [36] – see https://hpc.uni.lu, and the facilities provided by MediaGamma Ltd.

REFERENCES

[1] V. Krishna, *Auction theory*. Academic press, 2009.

[2] Y. Yang, R. Luo, M. Li, M. Zhou, W. Zhang, and J.Wang, "Mean field multi-agent reinforcement learning," *arXiv preprint arXiv:1802.05438*, 2018.

[3] R. Gummadi, P. Key, and A. Proutiere, "Repeated auctions under budget constraints: Optimal bidding strategies and equilibria," in *the Eighth Ad Auction Workshop*, 2012.

[4] K. Iyer, R. Johari, and M. Sundararajan, "Mean field equilibria of dynamic auctions with learning," *ACM SIGecom Exchanges*, vol. 10, no. 3, 2011.

[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*. ACM, 2012.

[6] W. Zhang, S. Yuan, and J. Wang, "Optimal real-time bidding for display advertising," in *KDD*. ACM, 2014.

[7] 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 *WSDM*. ACM, 2017.

[8] M. Du, R. Sassioui, G. Varisteas, M. Brorsson, O. Cherkaoui, and R. State, "Improving real-time bidding using a constrained markov decision process," in *ADMA*. Springer, 2017.

[9] 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.

[10] J. Jin, C. Song, H. Li, K. Gai, J. Wang, and W. Zhang, "Real-time bidding with multi-agent reinforcement learning in display advertising," *arXiv preprint arXiv:1802.09756*, 2018.

[11] J. Zhao, G. Qiu, Z. Guan, W. Zhao, and X. He, "Deep reinforcement learning for sponsored search real-time bidding," *arXiv preprint arXiv:1803.00259*, 2018.

[12] P. Hernandez-Leal, M. Kaisers, T. Baarslag, and E. M. de Cote, "A survey of learning in multiagent environments: Dealing with non-stationarity," *arXiv preprint arXiv:1707.09183*, 2017.

[13] M. L. Littman, "Markov games as a framework for multi-agent reinforcement learning," in *Machine Learning Proceedings 1994*. Elsevier, 1994.

[14] S. P. Choi, D.-Y. Yeung, and N. L. Zhang, "An environment model for nonstationary reinforcement learning," in *Advances in neural information processing systems*, 2000, pp. 987–993.

[15] J. Hu and M. P. Wellman, "Nash q-learning for general-sum stochastic games," *J. Mach. Learn. Res.*, vol. 4, pp. 1039–1069, Dec. 2003. [Online]. Available: http://dl.acm.org/citation.cfm?id=945365.964288

[16] K. Iyer, R. Johari, and M. Sundararajan, "Mean field equilibria of dynamic auctions with learning," *Management Science*, vol. 60, no. 12, 2014.

[17] K. Ren, J. Qin, L. Zheng, Z. Yang, W. Zhang, L. Qiu, and Y. Yu, "Deep recurrent survival analysis," *AAAI*, 2019.

[18] Y. Wang, K. Ren, W. Zhang, J. Wang, and Y. Yu, "Functional bid landscape forecasting for display advertising," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2016.

[19] W. C.-H. Wu, M.-Y. Yeh, and M.-S. Chen, "Predicting winning price in real time bidding with censored data," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2015, pp. 1305–1314.

[20] R. G. Miller Jr, *Survival analysis*. John Wiley & Sons, 2011, vol. 66.

[21] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017.

[22] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Acoustics, speech and signal processing (icassp), 2013 ieee international conference on*. IEEE, 2013.

[23] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," *arXiv preprint arXiv:1802.05365*, 2018.

[24] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[25] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding with unsupervised learning," Technical report, OpenAI, Tech. Rep., 2018.

[26] J. Hu and M. P. Wellman, "Nash q-learning for general-sum stochastic games," *Journal of machine learning research*, vol. 4, no. Nov, pp. 1039–1069, 2003.

[27] M. Huang, R. P. Malhamé, P. E. Caines *et al.*, "Large population stochastic dynamic games: closed-loop mckean-vlasov systems and the nash certainty equivalence principle," *Communications in Information & Systems*, vol. 6, no. 3, pp. 221–252, 2006.

[28] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[29] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

[30] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[31] W. A. Knaus, F. E. Harrell, J. Lynn, L. Goldman, R. S. Phillips, A. F. Connors, N. V. Dawson, W. J. Fulkerson, R. M. Califf, N. Desbiens *et al.*, "The support prognostic model: objective estimates of survival for seriously ill hospitalized adults," *Annals of internal medicine*, vol. 122, no. 3, pp. 191–203, 1995.

[32] H. J. and A. J. S., "Neural survival recommender," in *WSDM*, 2017. [Online]. Available: http://doi.acm.org/10.1145/3018661.3018719

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

[34] W. Zhang, S. Yuan, and J. Wang, "Real-time bidding benchmarking with ipinyou dataset," *CoRR*, vol. abs/1407.7073, 2014. [Online]. Available: http://arxiv.org/abs/1407.7073

[35] H. B. McMahan, G. Holt, D. Sculley, M. Young, D. Ebner, J. Grady, L. Nie, T. Phillips, E. Davydov, D. Golovin *et al.*, "Ad click prediction: a view from the trenches," in *KDD*. ACM, 2013.

[36] S. Varrette, P. Bouvry, H. Cartiaux, and F. Georgatos, "Management of an academic hpc cluster: The ul experience," in *Proc. of the 2014 Intl. Conf. on High Performance Computing & Simulation (HPCS 2014)*. Bologna, Italy: IEEE, July 2014.