Real-Time Bid Prediction using Thompson Sampling-Based Expert Selection

Elena Ikonomovska
Turn Inc.
835 Main St.
Redwood City, California
eikonomovska@turn.com

Sina Jafarpour
Turn Inc.
835 Main St.
Redwood City, California
sjafarpour@turn.com

Ali Dasdan
Turn Inc.
835 Main St.
Redwood City, California
adasdan@turn.com

ABSTRACT

We study online meta-learners for real-time bid prediction that predict by selecting a single best predictor among several subordinate prediction algorithms, here called "experts". These predictors belong to the family of context-dependent past performance estimators that make a prediction only when the instance to be predicted falls within their areas of expertise. Within the advertising ecosystem, it is very common for the contextual information to be incomplete, hence, it is natural for some of the experts to abstain from making predictions on some of the instances. Experts' areas of expertise can overlap, which makes their predictions less suitable for merging; as such, they lend themselves better to the problem of best expert selection. In addition, their performance varies over time, which gives the expert selection problem a non-stochastic, adversarial flavor. In this paper we propose to use probability sampling (via Thompson Sampling) as a meta-learning algorithm that samples from the pool of experts for the purpose of bid prediction. We show performance results from the comparison of our approach to multiple state-of-the-art algorithms using exploration scavenging on a log file of over 300 million ad impressions, as well as comparison to a baseline rule-based model using production traffic from a leading DSP platform.

Categories and Subject Descriptors

I.I [**Symbolic and Algebraic Manipulation**]: Applications; I.2 [**Artificial Intelligence**]: Learning—*Concept learning*; I.6 [**Simulation and Modeling**]: Applications—*Miscellaneous*

General Terms

Performance, Design, Experimentation

Keywords

Randomized probability matching, Bayesian online learning, Online algorithms, Online advertising, Multi-armed bandits

KDD'15, August 10-13, 2015, Sydney, NSW, Australia.

DOI: http://dx.doi.org/10.1145/2783258.2788586.

1. INTRODUCTION

With the increasing adoption of real time bidding (RTB), internet advertising models are undergoing a revolution. RTB enables marketers to participate in real time auctions for ad placement that occur seconds before a website loads. This functionality has helped to make advertising much more precise and personalized and enables advertisers to reach their target audiences. Programmatic trading, the technology behind the RTB, offers an immense potential for cost effective advertising. However, in order to realize this potential, advertisers need to intelligently adjust their bids as the market conditions fluctuate and real time feedback is received regarding the efficacy of a campaign. Demand-Side Platforms (DSPs) offer real time bid prediction [25, 18] to help advertisers find the optimal bid value in milliseconds. The goal of real time bid prediction is to maximize campaign performance goals under a budget constraint. Typical performance goals are minimizing cost-per-click (CPC) or cost-per-action (CPA), as well as maximizing click-through-rate (CTR) or action-rate (AR).

Predicting the performance of an ad is a challenging problem for many reasons. First of all, each decision to buy an impression and at what price needs to be made in just few milliseconds. Top DPSs typically receive a few million bid requests per second coming from users that are exploring the web all around the globe. Most of the time, the necessary information to make the optimal bidding decision is missing or delayed. For example, user data is based on cookies rather than real users. In addition, a significant number of visitors are likely to be entirely new with no historical consumption record whatsoever. In many web-based scenarios, the content universe undergoes frequent changes, with content popularity changing over time as well. The situation is similar with respect to advertisers' content (ads and campaigns); this is known as the cold-start problem.

The above problem is known as a feature-based exploration-exploitation problem [21] and to solve this problem we propose learning a pool of simple online prediction algorithms (predictors) with a partial view of the full contextual information. Specifically, for each triple of (user, advertiser, publisher) features we create a prediction strategy (an "expert") that examines only those features and makes predictions based on their values. In our framework the experts may choose to abstain from giving a prediction on a given instance. As in [6, 11], we shall call these experts *specialists*, because they are allowed to abstain when the instance doesn't fall into their areas of expertise, or specialities.

In this paper we shall deal with a pool-based real-time bid prediction. The learner is presented with a fixed pool of *specialists* that learn according to a modified online learning protocol with batch updates. Learning proceeds in a sequence of trials. On each trial the learner receives one instance from a fixed domain and feeds this instance to the *specialists* in the pool. The learner then selects one *specialist* from the pool and uses its estimate towards the bid computation. If the bid wins in the external auction then the learner pays a cost. After some time, the adversary provides the learner with the true labels and then the learner induces an updated set of *specialists* based on the newly received information.

We frame the problem of best specialist selection in the multi-armed bandit framework and propose to use randomized probability matching. An important component of the online learning framework is a loss function that associates a non-negative loss to each pair of prediction and outcome. In the multi-armed bandit framework, however, there is no obvious loss function defined on an instance basis.

Contributions. Our contributions are threefold. First, we describe how randomized probability matching can be naturally applied to settings like real-time bid prediction for online advertising. In particular, we describe a new version of a Thompson Sampling algorithm customized to suit such conditions [1]. It can be viewed as a meta-learning algorithm that selects from a pool of *experts*. Our work is one of the first in trying to use bandit type ideas for sampling from an ensembles of *experts* in real-time bid prediction with budget constraints, and we strongly believe that one can build extremely practical, yet very simple and scalable algorithms by understanding the interplay between multi-armed bandits and learning from a pool of experts. Second, we show that this approach is useful in practice, scales well, and is easily deployed at large-scale in real world industrial systems for real time ad allocation. Finally, we provide empirical support and comparison with several state-of-the-art algorithms using real world campaign data provided by Turn, a leading Demand-Side Platform (DSP).

2. BRIEF SYSTEM OVERVIEW

A simplified illustration of the advertising ecosystem and a typical ad call flow is given in Fig. 1. The figure illustrates a user with a web browser, an ad server, an ad exchange, a Data Management Platform (DMP) and a Demand Side Platform (DSP).

The ad call flow starts from the user's browser. The call is initiated when a user with **uid** opens a webpage at URL **url**. The user's browser then contacts the ad exchange the publisher is integrated with. The ad exchange appends **uid** and **url** to a request submitted to several DSP partners. Using the *uid* and *url*, DSPs return to their decision enginers the information about the given user and the web page that they can find in their stores. Each DSP runs an auction among the eligible ads (normally those that satisfy certain targeting constraints) from its list of existing campaigns. Then, one or more ads, depending on the space and layout characteristics, are chosen for display and ranked based on some function of expected performance and advertiser's bid.

[1]e.g. minimizing effective CPC (eCPC) and maximizing advertiser ROI.

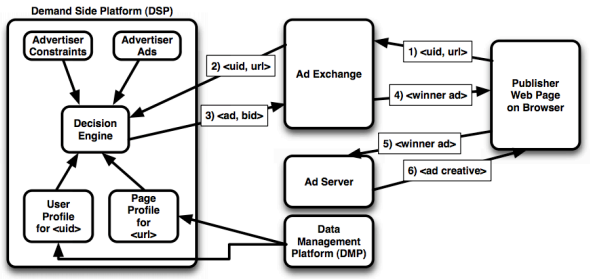

Figure 1: Ad call flow between a web browser and a demand side platform.

The bidding DSPs return back their proposed ad and bid pair back to the ad exchange. Once the ad exchange collects all the bids, it runs an auction (usually a second price auction) and determines the winning bid and the corresponding ad. It then passes the winning ad and the location of its creative back to the browser. The browser then collects the creative and finally returns the page to the user [19].

This entire flow needs to happen within a small portion of a second so that the user can see the page with ads as soon as the page appears on her web browser. This latency and throughput constraints put extreme time constraints on each bidding DSP and their decision engines. Our proposed approach is at the heart of the decision engine and works under such time constraints with a satisfactory performance.

3. PROBLEM FORMULATION

For the sake of clarity, we will only talk about click events, and focus on maximizing the CTR or minimizing the cost-per-click (CPC). Our methodology is equally applicable to other types of events such as conversion events, engagement events, etc. Moreover, without loss of generality, we will focus on the display channel. We provide a bottom-up formulation of the problem that provides the skeleton of our solution constructed on ideas such as: feature hierarchies, randomized probability matching, and sleeping experts.

3.1 Feature space and predictors

In the online advertising context, the feature space is usually constructed along three main dimensions: *user*, *publisher* represented as webpages, *advertisers* represented as ads. Time and other dimensions are extra. Let $\mathcal{U} = \{u_1, u_2, ..., u_l\}$, $\mathcal{P} = \{p_1, p_2, ..., p_m\}$, and $\mathcal{A} = \{a_1, a_2, ..., a_n\}$ represent all the users, webpages and ads (with creatives) in a DSP, respectively. The goal in bid prediction is then to find among all eligible ads the ad that has the highest probability of a click for a given user on a given webpage. One way to model this is to represent users, webpages, and ads using a set of explicit features and build a classification model. However, due to the sparsity of user-level data, an alternative and better idea is to compute the number of times an ad was displayed to "similar" users on a given website and observe how many of these impressions result in a click. Then, the CTR for this user can simply be estimated as the total number of clicks among all similar users divided by the total number of impressions. This kind of user grouping can either be achieved by explicit clustering based on some

similarity metric or it can be implicitly done by using data hierarchies [19].

For example, every ad in the DSP can be considered as belonging to an advertising campaign which in turn belongs to an advertiser (e.g., Advertiser: 'Acme Cars' → Campaign: '2011 Year End Sales' → Ad: 'Incredible Year End Sales Event!'). Similarly, a website on which an ad will be displayed is under a top level domain (TLD) which is owned by a publisher and the publisher itself might belong to some category based on its main content (e.g., Publisher Type: 'News' → Publisher: 'Acme City Times' → Page: 'Auto News'). If we assume that user, publisher, and advertiser data have l_u, l_p, l_a levels in their respective data hierarchies, there are $l_u \times l_p \times l_a$ possible ways of combining count data for a given ($user \times publisher \times advertiser$) triple.

Using these various levels of count data aggregated along the time dimension we can create simple context-dependent predictors that are able to produce a maximum likelihood estimate of the CTR. Experts that use higher-level data will be susceptible to high bias in their estimates. On the other hand, experts that use lower-level data will be susceptible to high variance, due to a limited number of data points from which the estimates are drawn. Naturally, by observing more data points in the lower levels of the data hierarchies the corresponding experts will become more accurate over time. In the next section, we discuss how we can combine these estimators to obtain a final prediction.

3.2 Meta-learning

Let let $\hat{x}^1_{ijk}$, $\hat{x}^2_{ijk}$, ..., $\hat{x}^K_{ijk}$, denote the full set of maximum likelihood estimates of the CTR for user u_i, webpage p_j and ad a_k, at K different level combinations. For clarity, we will drop the indexes and use $\hat{x}^1$, $\hat{x}^2$, ..., $\hat{x}^K$ instead. Recall that each of these levels corresponds to a distinct combination of the user, publisher and advertiser hierarchies and, as such, it is not always clear which one will yield the best estimate of the true CTR p_{ijk}.

Most of the time user-publisher-advertiser data will be missing, and the estimators using these information sources simply will not be computed. For example, a user IDs might not be found in the user profile servers, or the publisher's webpage will not match any of the known categories in the publisher taxonomy, or the advertiser has no past-performance data in the system. Moreover, there may not be sufficient number of click events in one of the hierarchical levels to calculate a reliable estimator output using the past performance observations. As a result, the number of available estimates will change over time, and in some cases might be zero.

One way to aggregate these estimates is to use a learning method that combines the available estimates into a single prediction. For example, Lee et al. [19] have used logistic regression to combine experts estimates for action rate prediction. Further, there exist a number of online learning algorithms designed for the scenario of missing experts from a pool of learners, such as the online SBayes [11]. However, due to the dynamic nature of the problem and the time-varying performance of the experts, these approaches are not most suitable. Rather than trying to combine multiple predictions, in this work we try to pick the best estimator among the K experts. The reasoning behind this decision is two-fold: First, through an online evaluation we have observed that under specific contextual decisions, the experts are able to provide very good estimates of the CTR without merging; second, their performance varies significantly over time. Hence, our problem can be formulated as an online expert selection problem, where the set of available experts changes over time.

3.3 Performance-based bid prediction

At an abstract level, advertisers place their bids in the form of CPC or CPA goals. In pure second-price auctions the dominant bidding strategy for the advertisers is to submit their private true value [10]. Therefore, the value of the impression is typically calculated with the probability of a click (predicted by DSPs) multiplied by the value of the click (given by advertisers). The resulting bid for a given impression at time t would be calculated as:

$$Bid(t) = \theta(t) \times goal$$

where the goal is a fixed number w.r.t. the given campaign that amounts to the monetary value the advertiser assignes to a click, and $\theta(t)$ is the click probability. In practice, however, this is never the case. Besides the additional bid adjustments which are done post-CTR computation, the advertisers can further influence the bid price through multiple scaling factors.

Therefore, when minimizing the eCPC, one needs to take into account the expected cost for the expected number of clicks. In other words, minimizing the eCPC is not equivalent to the CTR maximization strategy. To illustrate this, let's assume that we need to minimize the eCPC, and the total budget is $1000. Assume further that we have two different prediction strategies whose average estimates for the CTR are: 12×10^{-3} and 20×10^{-3} correspondingly. Due to the difference in the expected values of these estimates and the inherent sampling biases, the bid price would be different though not linearly dependent, so let's assume that the average cost per impression is $0.05 and $0.12 correspondingly. That means one could buy on average around 20,000 impressions in the first case and around 8,334 impressions in the second case. This leads us to about 200 clicks with the first strategy and 175 with the second, and correspondingly $5 and $5.7 effective cost per click (eCPC). In this case, using the first strategy whose CTR estimate is lower can lead us to a better eCPC.

4. MULTI-ARMED BANDITS

Choosing the best expert for bid prediction can be naturally viewed from the perspective of exploration-exploitation trade-off. The concept of exploration-exploitation is central to problems in decision making under uncertainty, and is best illustrated by the multi-armed bandit (MAB) problem.

In its simplest formulation (generally referred to as stochastic), a bandit problem consists of a set of K probability distributions $< D_1, ..., D_K >$ with associated expected values $< \mu_1, ..., \mu_K >$. Initially, the distributions are unknown to the player. In fact, these distributions are generally interpreted as corresponding to arms on a slot machine; the learning algorithm is viewed as a gambler whose goal is to collect as much money as possible by pulling these arms over many turns. At each turn, $t = 1, 2, ...$, the player selects an arm, with index a_t, and receives a reward $y_t \sim D_j(t)$. The goal of the algorithm is to find out which distribution has the highest expected value, and also to gain as much reward as possible during the course of the game.

The added complexity of using bandit algorithms in bid prediction problems is two-fold: First, we assume that the bandit algorithm has access to a set of experts $S_t \in 1,\ K$ whose performance can change over time; second, for each turn t the bandit has to pay an unknown variable cost c_t associated with winning the auction. Hence, each trial of the bid prediction problem is a trial between the algorithm and an adversary in a game that consists of the following steps:

1. The adversary choses a set $S_t \subseteq \{1, ..., K\}$ of experts that are available at iteration t.
2. Each available expert $i \in S_t$ provides an estimate $\hat{x}^i$.
3. The algorithm chooses one expert s_j from S_t and uses its estimate to compute a bid.
4. The adversary chooses a cost c_t for buying that impression and a binary outcome $y_t \in \{0, 1\}$ (no click / click).
5. The algorithm pays a cost c_t.

Various strategies have been proposed that maximize the expected reward, but very few consider the inherent cost for playing an arm. As we showed, an algorithm that strives to maximize the expected reward (CTR) can be sub-optimal in minimizing the effective cost-per-click (eCPC). We propose a solution based on *randomized probability matching*, better known as Thompson Sampling [27] (TS). In the next section we provide the details of our solution.

4.1 Randomized probability matching

Randomized probability matching plays each action *randomly* in proportion to its probability of being *optimal*. This approach is known as *Thompson Sampling* and has been shown to be especially successful in systems whose limitations allow only periodic (batch) updates. It is broadly applicable, easy to implement and can be extended to work with a broad class of reward distributions.

We consider a general probabilistic, or Bayesian, formulation in which uncertain quantities are modeled as random variables. Let $\mathbf{y}_t = (y_1, ..., y_t)$ denote the sequence of rewards observed up to time t, such that $y_i \in \{0, 1\}, \forall i = 1, ..., t$. We focus on binary rewards because clicks are binary events. Let a_t denote the played arm at time t. The success of each arm is modeled as an IID random variable, Y_1, Y_2, ..., Y_K living in a space $\mathcal{Y} = \{0, 1\}^K$. Since each random variable has a binary outcome, each arm is modeled with a Bernoulli distribution with success probability θ_a. It follows that each y_t was generated independently from the reward distribution $f_{a_t}(y|\theta)$ of the played arm, where θ is an unknown parameter vector that governs the success probabilities of the Bernoulli distributions of the arms.

Let $\mathbf{c}_t = (c_1, ..., c_t)$ denote the sequence of costs that the algorithm has payed for each observed reward up to time t, such that $c_i \in \mathbb{R}, \forall i = 1, ..., t$. The cost of each arm is modeled as an independent and identically distributed random variable, C_1, C_2, ..., C_K defined in a space $\mathcal{C} = \mathbb{R}^K$. Now, the cost distribution of each arm is modeled with a Gaussian distribution with parameters m_a and σ_a. It follows that each c_t was generated independently from the cost distribution $f_{a_t}(c|m, \sigma)$ of the played arm, where m and σ are unknown parameter vectors.

Let $\mu_a(\theta) = E(y_t|\theta, a_t = a)$ denote the expected reward coming from the reward distribution $f_{a_t}(y|\theta)$. Let $\nu_a(m, \sigma) = E(c_t|m, \sigma, a_t = a)$ denote the expected cost coming from the cost distribution $f_{a_t}(c|m, \sigma)$. If our goal was to maximize the number of clicks without considering the price we are paying for each impression, then the optimal long run strategy would be to always choose the arm with the largest $\mu_a(\theta)$. However, our goal is to minimize the ratio of the total cost and the total number of clicks, i.e., the cost per click. Therefore, we will denote with $\mu_a(\theta, m, \sigma)$ the expected cost per click coming from the reward distribution $f_{a_t}(y|\theta)$ and the cost distribution $f_{a_t}(c|m, \sigma)$.

Let $p(\theta)$ denote a prior probability on θ, and $p(m, \sigma)$ denote a prior probability on m and σ. Assuming that we have a method to compute $\mu_a = \mu_a(\theta, m, \sigma)$, according to the principles of randomized probability matching one can compute the initial allocation probabilities:

$$w_{a,0} = P(\mu_a = \min\{\mu_1, \mu_2, ..., \mu_K\}). \tag{1}$$

Eq. 1 can be expressed as an integral of an indicator function. Let $I_a(\theta, m, \sigma) = 1$ if $\mu_a(\theta, m, \sigma) = \min\{\mu_1(\theta, m, \sigma), ..., \mu_K(\theta, m, \sigma)\}$, and $I_a(\theta, m, \sigma) = 0$ otherwise. Then

$$w_{a,0} = E(I_a(\theta, m, \sigma)) = \int I_a(\theta, m, \sigma)p(\theta)p(m, \sigma)d\theta dm d\sigma. \tag{2}$$

Here we assume that the success variables and the cost variables are independent from each other. The prior distributions represent our beliefs or a-priori knowledge about the success probability of each arm as well as the cost parameters per arm. As rewards and costs from the bandit process are observed, the parameters of the reward and cost distributions are updated through the process of Bayesian updating. After observing the sequence of rewards $\mathbf{y}_t$ and costs $\mathbf{c}_t$ at time t the posterior distribution of θ is

$$p(\theta|\mathbf{y}_t) \propto p(\theta) \prod_{\tau=1}^{t} f_{a_t}(y_\tau|\theta) \tag{3}$$

while the posterior distribution of m and σ is

$$p(m, \sigma|\mathbf{c}_t) \propto p(m, \sigma) \prod_{\tau=1}^{t} f_{a_t}(c_\tau|m, \sigma). \tag{4}$$

Hence, to compute the allocation probabilities in the next iteration all we need is a component that will be able to estimate the expected reward μ_a and up-to-date beliefs represented with the posterior distributions $p(\theta|\mathbf{y}_t)$ and $p(m, \sigma|\mathbf{c}_t)$:

$$\begin{aligned} w_{a,t} &= P(\mu_a = max\{\mu_1, \mu_2, ..., \mu_K\}|\mathbf{y}_t, \mathbf{c}_t) \\ &= E(I_a(\theta, m, \sigma)|\mathbf{y}_t, \mathbf{c}_t). \end{aligned} \tag{5}$$

Eq. 1 and Eq. 5 can be computed using the law of large numbers by simulation. Let $\theta^{(1)}$, ..., $\theta^{(G)}$ be a sample of independent draws from $p(\theta|\mathbf{y}_t)$, and $(m^{(1)}, \sigma)$, ..., $(m^{(G)}, \sigma)$ be a sample of independent draws from $p(m, \sigma|\mathbf{c}_t)$. By the law of large numbers,

$$w_{a,t} = \lim_{G\to\infty} \frac{1}{G} \sum_{g=1}^{G} I_a(\theta^{(g)}, m^{(g)}, \sigma). \tag{6}$$

Eq. 6 estimates the allocation probabilities $w_{a,t}$ by the empirical proportion of Monte Carlo samples in which $\mu_a(\theta^{(g)}, m^{(g)}, \sigma)$ is maximal. Choosing adequate conjugate prior distributions $p(\theta)$ and $p(m, \sigma)$ makes sampling independent draws

of θ and m possible. In our case we will be working with a Beta distribution as a prior for the Bernoulli distribution used to model the reward likelihood, and a Gaussian prior for the Gaussian distribution used to model the cost likelihood.

Estimating $\mu_a(\theta, m, \sigma)$ is a separate problem which will not be discussed here in full detail. The method simply estimates the expected cost per click by the method of simulation, using the posterior distributions $p(\theta|\mathbf{y}_t)$ and $p(m, \sigma|\mathbf{c}_t)$. Since posterior draws are all that is needed to compute the allocation probabilities one can apply randomized probability matching with almost any family of reward distributions.

4.2 Thompson Sampling with double priors

The pseudo-code of our proposed Thompson Sampling algorithm is given in Algorithm 1. The algorithm starts from its prior beliefs on the expected success rate and the expected cost, implemented with a Beta and a Gaussian prior distribution, respectively. In each trial, the algorithm first observes the set of available experts S_t and their estimates $\hat{\mathbf{x}}_t^j$ (line 3), and performs a Monte Carlo simulation to calculate the allocation probabilities using Eq. 6 (line 4). The simulation consists of drawing independent samples from the posterior distributions of θ and m. Then the algorithm chooses to play an expert according to the allocation probabilities $\mathbf{w}_t$ (line 5), and uses its estimate of the CTR to calculate the bid. If the bid wins the external auction, the algorithm observes the outcome y_t and the cost c_t (line 8), and updates the posterior distributions by updating their parameters (lines 10-14).

Algorithm 1 Thompson Sampling with double priors

1: Initialize S(1) = 0, F(1) = 0, , T(1)=0, $\hat{m}(1) = 0$
2: **for** $t = 1, 2, ..., T$ **do**
3: Observe S_t and $\hat{\mathbf{x}}_t^j$
4: For $i \in S_t$ calculate the allocation probabilities $w_{i,t}$ using $Beta(S_a(t) + 1, F_a(t) + 1)$ and $\mathcal{N}(m(t), \sigma(t)^2)$
5: Choose one expert a at random according to $\mathbf{w}_t$
6: Use estimate $\hat{x}_t^a$ to submit a bid b_t
7: **if** b_t won the auction **then**
8: Observe the outcome y_t and the cost c_t
9: Update the posterior distributions of expert a:
10: $S_a(t+1) = S_a(t) + y_t$
11: $F_a(t+1) = F_a(t) + 1 - y_t$
12: $m(t+1) = \frac{\sigma_0^2}{\frac{\sigma^2}{t}+\sigma_0^2} c + \frac{\sigma^2}{\frac{\sigma^2}{t}+\sigma_0^2} m_0$
13: $\sigma(t+1)^2 = \left(\frac{1}{\sigma_0^2} + \frac{t}{\sigma^2}\right)^{-1}$
14: $T_a(t+1) = T_a(t) + 1$
15: **end if**
16: **end for**

Using Beta(α, β) priors is useful for Bernoulli rewards because the Beta distribution is a conjugate prior distribution for Bernoulli. This enables simple Bayesian updates through updating the parameters α and β. At time t having observed $S_a(t)$ successes and $F_a(t)$ failures in $T_a(t)$ plays of arm a, the posterior distribution is Beta($S_a(t)$+ y_t, $F_a(t)$ + 1 - y_t).

Similarly, the conjugate prior distribution for a Gaussian likelihood distribution is Gaussian, which again enables simple updates through the parameters of the distribution. We choose to model the likelihood of the cost $f_a(c|m, \sigma)$ with a Gaussian distribution $\mathcal{N}(m, \sigma^2) \sim \frac{1}{\sigma^t} \exp\left(-\frac{1}{2\sigma^2}\sum_i (c_i - m)^2\right)$.

By choosing a conjugate prior distribution $\mathcal{N}(m_0, \sigma_0^2)$s:

$$p(m_0, \sigma_0) \propto \frac{1}{\sigma_0} \exp\left(-\frac{1}{2\sigma_0^2}(m - m_0)^2\right)$$

where the prior mean m_0 is typically chosen to be 0, and the prior variance σ_0 is some large value, the resulting posterior distribution $p(m|\mathbf{c}_t)$ is obtained by simply multiplying the likelihood $f_a(c|m, \sigma)$ and the prior $p(m_0, \sigma_0)$:

$$\mathcal{N}\left(\frac{\sigma_0^2}{\frac{\sigma^2}{t}+\sigma_0^2} c + \frac{\sigma^2}{\frac{\sigma^2}{t}+\sigma_0^2} m_0, \left(\frac{1}{\sigma_0^2} + \frac{t}{\sigma^2}\right)^{-1}\right) \quad (7)$$

Both of the distributions can be updated using batch updates of their parameters.

5. NON-STATIONARY SYSTEMS

In a variety of practical applications as well as in our system the time evolution of the system, and in particular of the reward distributions, is gradual. Actually, if the relationship between the input features and the target variable is stationary then we can confidently predict that the reward distributions for each cold-started arm will be evolving in a similar way. As the estimates of the experts improve over time we should expect that there will a gradual increase in the running CTRs and a shift towards convergence. All of the experts will be characterized with an initial period of high instability and uncertainty. An example of how one should expect the evolution of the CTR would look like is given in Fig. 2.

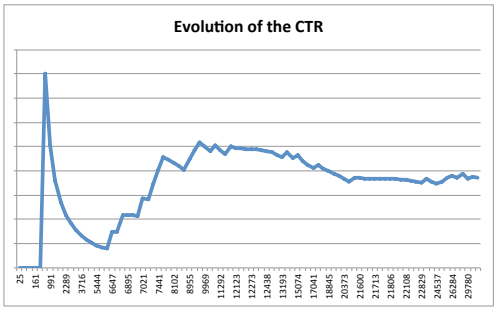

Figure 2: Evolution of the click-through-rate (CTR) for one specialists w.r.t. a given campaign.

Hence, for dynamic multi-armed bandit problems one should prefer policies that take into account the fact that the reward distributions change in a gradual manner, starting from a point of high instability and uncertainty and moving slowly towards a more stable phase. While Thompson Sampling naturally handles this evolution, it can be easily adapted to track potential changes.

5.1 Time-varying reward distributions

As discussed previously the problem we are solving falls in the family of multi-armed bandit problems with non-stationary reward distributions, where the optimal arm changes over time. We have discovered two sources of change in the reward distributions of the experts (arms):

1. Due to the slow increase in the accuracy and prediction power of a cold-started expert i, we expect an

initial period of instability where rewards arrive in an adversarial manner, followed by a gradual increase in the probability of successes S_i obtained in T_i number of Bernoulli trials and a saturation phase.

2. Due to the dynamic nature of the bidding process and the possibility of a change in the bidding landscape or the setup of the campaign, experts that performed well in the past might start to underperform compared to others w.r.t. a given campaign.

To be able to track changes in the probability of success of each expert, we adopt an approach based on exponential smoothing proposed by Gupta et al. [14]. This is a filtering technique that can be easily and efficiently implemented in a large-scale system like ours. The gist of exponential smoothing is exponential weighting of the outcomes in each trial, such that older outcomes get smaller weights and hence contribute less to the current estimate of the success probability θ_i. This is a typical forgetting mechanism used in many online learning algorithms designed for dynamic environments, and has been used for computing prequential (predictive-sequential) error estimates as well, where the sum of losses $L_A(t)$ at time t is computed as:

$$L_A(t) = \alpha L_A(t-1) + L_A(\hat{y}_t, y_t)$$

where $\alpha \in [0, 1]$ is a fading factor that determines the speed with which past errors will be diminished.

As discussed previously the posterior of the reward distribution at time t of expert a is modeled with a Beta$(S_a(t)+y_t, F_a(t) + 1 - y_t)$ distribution, where $S_i(t)$ and $F_i(t)$ denote the successes and the failures up to time t. Exponential smoothing is simply implemented through a new set of update rules on the parameters of the Beta distributions S_i and F_i, $\forall i = 1, ..., K$ and a threshold parameter C which determines the upper bound on the variance of the Beta distribution:

1. If $S_a(t-1) + F_a(t-1) < C$

$$\begin{aligned} S_a(t) &= S_a(t-1) + y_t \\ F_a(t) &= F_a(t-1) + 1 - y_t \end{aligned} \tag{8}$$

2. If $S_a(t-1) + F_a(t-1) \geq C$

$$\begin{aligned} S_a(t) &= (S_a(t-1) + y_t)\frac{C}{C+1} \\ F_a(t) &= (F_a(t-1) + 1 - y_t)\frac{C}{C+1} \end{aligned} \tag{9}$$

The effect of these update rules are that: 1) they ensure that the sum of $S_a(t)$ and $F_a(t)$ never grows above C; 2):

$$\mu_a(t) = \frac{S_a(t)}{S_a(t) + F_a(t)} = \alpha\mu_a(t-1) + (1-\alpha)y_t \tag{10}$$

where $\alpha = \frac{C}{C+1}$; and 3):

$$0 \leq \sigma_a^2(t) \leq \frac{1}{4(C+1)} \tag{11}$$

The direct effect of bounding the variance of the Beta distribution through the parameter C is to enabling indefinite exploration and through that, detection of relative changes in the expected values of the reward distributions of the experts. Due to the scale of our system and the large number of campaigns with different pacing parameters, the value of the parameter in practice needs to be set individually.

Exponential smoothing, however, is not sufficient to capture the dynamics of the reward distributions that are specific to cold-started experts. For that purpose, our online learning samplers operate on sliding windows. Using a sliding window enables us to discard the counts of successes and failures from the early stages of the specialized learners and maintain estimates using the most recent data.

5.2 Extensions: Factoring in Context

In the traditional, non-contextual multi-armed bandit problem, the learner has no access to arm features and simply competes with pulling the best of K arms in hindsight. As opposed to the traditional K-armed bandit problems, features of the arms may be useful to infer the conditional average payoff of an arm and improve the total average payoff over time. Our idea is to use real-time performance features of each expert, and time decaying functions that model the uncertainty in their performance in the early stages of their learning process as contextual information.

We use a predefined set of performance features and the time span w.r.t. the number of times the specialist was given a chance to learn: 1) the empirical estimate of the success probability $f_1(t) = S_a(t)/T_a(t)$, the running average of the specialist's predictions $f_2(t) = \frac{1}{T_a(t)}\sum_{\tau=1,a_\tau=a}^{t} \hat{y}_{a,\tau}$, the running variance of the specialist's predictions $f_3(t) = \frac{1}{T_a(t)}\sum_{\tau=1,a_\tau=a}^{t}(\hat{y}_{a,\tau} - \bar{\hat{y}}_{a,\tau})^2$, the submitted bid $f_4(t) = b_t$, and the time-decaying function $f_5(t) = \frac{1}{\sqrt{T_a(t)+1}}$.

The most notable example of bandits with side information are contextual bandits with linear payoff functions, which is a well studied problem in statistics and machine learning. Agrawal and Goyal have designed the first extension of Thompson Sampling for the case of the stochastic contextual multi-armed bandit problem, using Gaussian priors and Gaussian likelihood model [3]. We have adopted the same design of a contextual Thompson Sampling algorithm without any changes and used it for CTR estimation.

6. PRACTICAL ISSUES

The practical deployment of the Thompson Sampling algorithm involves the design and implementation of an offline and an online component. The offline component is designed to collect performance data which is being stored and accessed periodically, using Apache Hadoop and Pig. The online component, on the other hand, is executed by the ad servers in real time, and involves the process of sampling from the posterior distribution and calculating the allocation probabilities.

To ingest the data from the outside world, our data processing pipeline defines a process of transformation, join, and compression stages. Raw web logs need to be first synced to a designated location on the Hadoop file system, where they would be picked up by the modules for click attribution. This stage is followed by click de-duplication and fraud detection, and at last the aggregated data becomes accessible for querying. Because of this complicated pipeline, updates to the learners are periodical and can only happen in batches. It becomes clear that algorithms that cannot operate with batch updates cannot be implemented in our platform.

Further, having different campaigns with different budgets, bidding pace, bidding tactics, goals and targeting criteria, the expert selection problem needs to be carried out

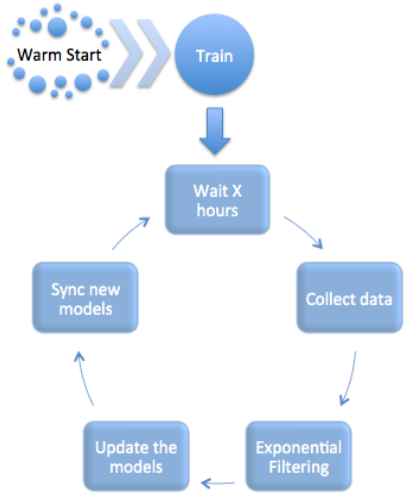

Figure 3: Illustration of the learning flow on HDFS.

at a more granular level. Therefore the data collection and model generation is done on a campaign level. For each campaign and for each expert that has been played, the data collection flow consists of a cyclic repetition of the following stages: collection and aggregation, exponential smoothing, generation of new hyper-parameters, as shown in Figure 3. The resulting model is in essence a key-value map, where the key is a combination of a campaign id and expert id, and value is a list of multiple summary statistics and distribution parameters.

After the new model is generated, it is synced to the 1000s of ad servers on the distributed platform, and its execution proceeds without real-time updates. It is worth mentioning that sampling from a Beta and a Gaussian distribution can be time consuming especially when the whole bid prediction process has to be done within few milliseconds. In order to speed up the sampling, we have used a fast implementation of the Mersenne-Twister algorithm [23] as a replacement of the random sampling function available in Java, as well as an implementation of the Ziggurat algorithm for generating random numbers from a Gaussian distribution [22]. Although using these improved samplers within the algorithms for sampling from a Beta and Gaussian distribution improved the sampling time significantly, due to the extremely sensitive time restrictions for some cases we also used precomputed samples and estimates.

7. RELATED WORK

The exploration / exploitation dilemma is an old problem and has received a significant amount of attention from the statistics and machine learning communities. The Multi-armed bandit problem has close connections to the Boosting method in classification [29], and to sparse recovery and compressed sensing [28]. Various strategies have been proposed throughout the decades, probability matching, greedy, hybrid strategies, methods based on indices such as the upper confidence bounds (UCB) and Gittins indices [12]. Among the oldest heuristics are the *randomized probability matching* or *posterior sampling* strategies. The first version of this Bayesian heuristic is around 80 years old, dating back to Thompson [27].

Thompson Sampling (TS) has only recently been established as a top performer for MABs with Bernoulli distributed rewards, and the reason for this delay has been the lack of theoretical understanding. Several studies [13, 26, 8, 24, 15] have empirically demonstrated the efficacy of TS. Weak guarantees have been provided by [13, 24] with a bound of $o(T)$ on the expected regret in time T. Some significant progress has recently been made by [1, 15, 2], who provided optimal regret bounds on the expected regret; Agrawal and Goyal provided high probability, near-optimal regret bounds for stochastic contextual bandits with linear payoffs [3].

It is useful to mention that one could imagine trying to solve the best specialist selection problem using EXP3- or EXP4-type approaches [5]. In EXP4-type approaches the player's goal is to combine the advice of the experts in such a way that its return is close to that of the best expert. EXP4 stands for "Exponential-weight algorithm for Exploration and Exploitation using Expert advice", hence, it is an algorithm that learns how to mix the probability distributions coming from N experts in order to generate a final mixture distribution over the set of arms of size K. It can be used in a simplified scenario, where each expert corresponds to a deterministic policy that maps a context to only one arm with probability 1.0, and with probability 0.0 to the rest of the arms. EXP3 is in essence a special case of EXP4 for this simplified scenario.

While in the expert-learning framework, each expert corresponds to a contextual policy for arm selection, in our setup experts are not a mapping from a context to an arm, but they are specialized arms whose outputs are the conditional probabilities $f_i(y_t|\theta)$. EXP3- and EXP4-type approaches cannot be easily modified towards more complex reward distributions or optimization goals. Therefore, none of these approaches provide any advantage over Thompson Sampling. Posterior sampling can be applied to a much broader class of problems, and one of its greatest strengths is its ability to incorporate prior knowledge in a flexible and coherent way.

Recently Li [20], motivated by the connection between Thompson Sampling and exponentiated updates, has designed a new family of algorithms called Generalized Thompson Sampling in the expert-learning framework [7]. Generalized Thompson Sampling (GTS) is very similar in structure to EXP4, with the difference of a more general update rule that uses a loss function to adjust the expert's weights. Each expert represents a greedy policy with respect to the probability of success (reward prediction) that maps a context to an arm. As in all existing analysis for Thompson Sampling, the assumption is that one of the experts correctly predicts the expected reward (probability of success). The generalization involves two loss functions: Logarithmic loss and square loss, for which regret bounds are derived. The author shows that Thompson Sampling is a special case of GTS when the logarithmic loss is used [2].

Due to the complexity of the performance-based bid prediction problem, very few algorithms are designed to maximize the expected reward (CTR) having into account budget limitations or costs for playing the arms. To the best of our

[2]The loss function is used to measure how well the expert predicts the average reward, given the context and the selected arm. In general, the loss function and the reward may be completely unrelated.

knowledge, there is only one work that treats the bandit problem with budget constraint and variable costs [9]. The authors propose UCBBV2 an UCB-type algorithm whose expected regret depends on the budget which constrains the total number of pulls. While this algorithm does take into account the variable costs per arms, it is not designed to work with batch updates and as such it is not appropriate for our platform.

8. EVALUATION METHODOLOGY

Our proposed framework of bid prediction has been implemented, tested, and deployed at Turn, a leading DSP. In this section, we present experimental results from our testing environment where we compared different strategies for expert selection. In addition, we also show results from real campaigns that serve large amounts of daily impressions in order to demonstrate the overall performance improvement in terms of CTR, eCPC, and ROI performance metrics. Advertiser's return of investment (ROI) is the total value ($\#click \times value$) divided by the total cost incurred ($\sum_i c_i$).

8.1 Experiments

The first part of our experimental evaluation consists of an offline policy evaluation of the proposed Thompson Sampling algorithm and its extension, the Contextual Thompson Sampling algorithm with a number of state-of-the-art multi-armed bandit strategies. The offline evaluation is performed on a large collection of over 300 million ad impressions, collected by Turn's processing pipeline in a period of 30 days. The impressions belong to 7 top spending advertisers that are running several campaigns on the platform. The second part of our experimental evaluation is an online comparison of Thompson Sampling on production data to our existing baseline algorithm through fair A/B testing. The baseline algorithm is a static, rule-based model that chooses the best expert in a deterministic manner.

First, we would like to emphasize the volatility of the CTR in few top performing estimators through the daily evolution of the CTR shown on Fig. 4. This CTR is computed by ordering our historic impressions based on the time stamp the impression was shown to a user and aggregated by day. We can see that the daily CTR varies significantly, which has strongly discouraged us from running experiments on simulated data.

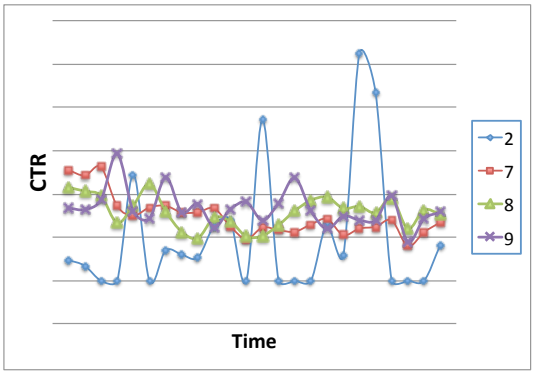

Figure 4: Illustration of the evolution of the CTR per day for the top performing experts.

In the comparison we have used the following state-of-the-art MAB algorithms: Chernoff UCB is an index based UCB-style algorithm with a tight upper confidence bound derived from the Chernoff bound ([16], page 278), EXP3 and EXP3.S [5] are bandit algorithms with experts advice, UCB1 and Tuned UCB1 [4], UCBBV2 [9]. For a fair comparison we ran all of the algorithms without using any explicit change detection or blind concept drift management. All of the algorithms were further tuned for best performance.

As a DSP, we do not have the luxury of executing arbitrary policies on real traffic, and the only viable alternative is to do offline policy evaluation. Offline policy evaluation is the process of evaluating a new strategy for behavior, or policy, using only observations collected during the execution of another policy. The difficulty of this problem stems from the lack of control over available data. In our case we are bounded to use the existing logged impressions and the corresponding choices of the baseline algorithm, which is our exploration policy.

Exploration Scavenging (also known as *offline replay* [17]) is a popular method for offline evaluation of new MAB policies π_i, by using the logged decisions of an existing baseline policy s. This method suggests to calculate an expected reward of a new policy s using the following equation:

$$\hat{R}(\pi) = \frac{1}{T}\sum_{t=1}^{T} r_{s(x_t)} \mathbf{1}[s(x_t) = \pi(x_t)] \sum_{a \in \mathcal{A}} \mathbf{1}[s(x_t) = a] w_{x_t,a}, \tag{12}$$

where T denotes the total number of historic logs used in the replay, A is the set of all possible actions (arms), x_t is the context for impression at time t, $\mathbf{1}$ is the indicator function, and $w_{x,a}$ is a normalization weight calculated from the whole set of impressions with contextual information x as

$$w_{x,a} = \frac{\sum_{t'=1}^{T} \mathbf{1}[x_{t'} = x] \times \mathbf{1}[\pi(x_{t'}) = a]}{\sum_{t'=1}^{T} \mathbf{1}[x_{t'} = x] \times \mathbf{1}[\pi(x_{t'}) = a] \times \mathbf{1}[\pi(x_{t'}) = s(x_{t'}]}.$$

Exploration scavenging first estimates the average reward of each action from the set of impression on which both the evaluation π and the exploration s policies agree on choosing the same action a, then applies the estimate to all the impressions for which the evaluation policy π suggests the action a. Fig. 5 shows the total scavenged CTR for each algorithm computed at the end of the run.

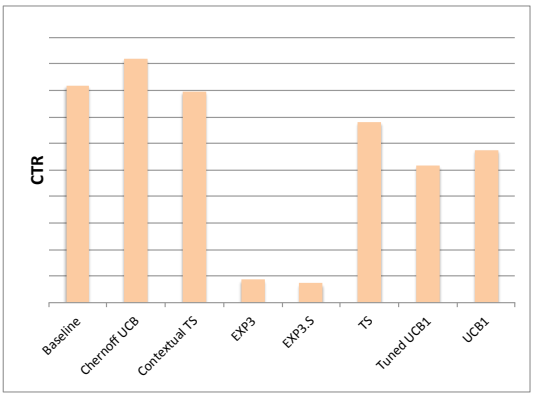

Figure 5: Total scavenged CTR per algorithm.

Beside exploration scavenging we also show results computed using only the impressions on which the evaluation

policy agreed with the exploration policy, without applying precomputed estimates of the average reward. We will refer to this approach as the matched CTR. Unlike the exploration scavenging estimates this approach will provide rather optimistic estimates of the CTR. These estimates are highly dependent on the number of matched impressions between the evaluation and the exploration policy. From all algorithms, TS and Contextual TS had the largest overlap with the baseline policy, with Contextual TS having twice as much more matched decisions than TS. Fig. 6 shows a comparison of the total matched CTR for all algorithms at the end of the run.

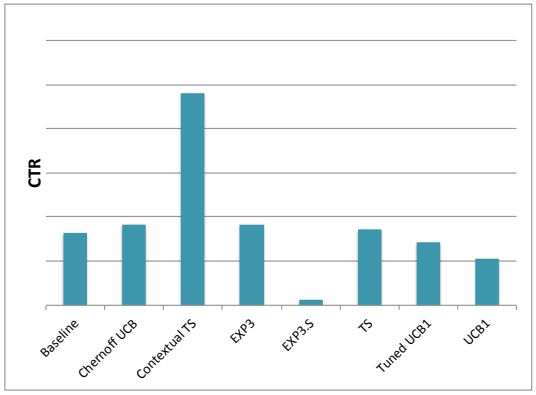

Figure 6: Total matched CTR per algorithm.

At last, Fig. 7 shows relative results in terms of the total ROI at the end of the run, which is our main metric of interest. The top leading algorithms are evidently TS and Contextual TS, with UCBBV2 and the baseline being third. As discussed previously, to work properly, UCB-type of algorithms require updates of the arm's upper confidence bounds instantaneously due to the deterministic indexes. In practice, this is rarely doable. On the other hand, Thompson sampling relies purely on random distribution sampling, and can maintain a reasonable balance between exploration and exploitation in each round while using a batch-updates.

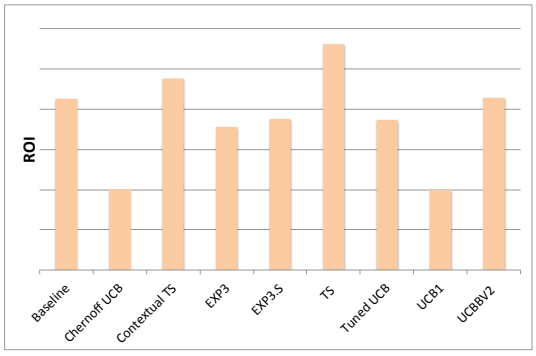

Figure 7: Total ROI per algorithm.

Due to the dynamic behavior of the expert's performance it is important to understand how the performance of each of the top algorithms evolves over time. Due to space limitations we will only show the evolutions of the CTR and the ROI for the Thompson Sampling algorithm and our baseline model evaluated on production traffic over the timeline of one day. Figure 8 and Figure 9 show summaries of the cumulative CTR and ROI on a global level as time proceeds. We can see that Thompson Sampling performs better than the baseline model both in terms of CTR and in terms of global ROI. Similarly, Figure 10 compares the cumulative eCPC of the two models as a function of time. As the plot indicates, TS is able to provide a lower overall eCPC compared to the baseline model.

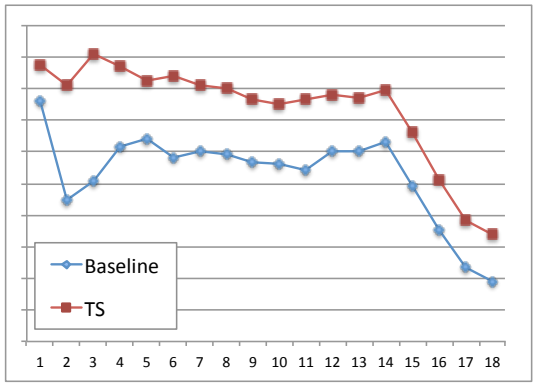

Figure 8: Evolution of CTR on production traffic.

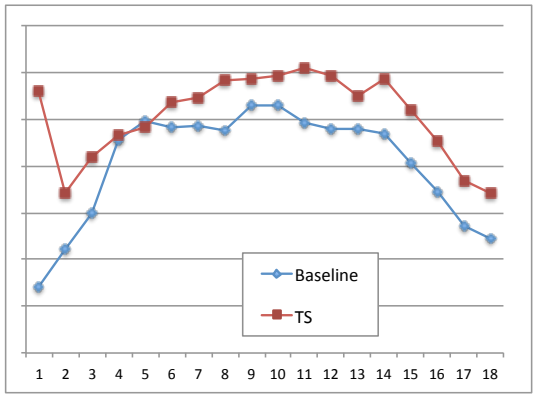

Figure 9: Evolution of ROI on production traffic.

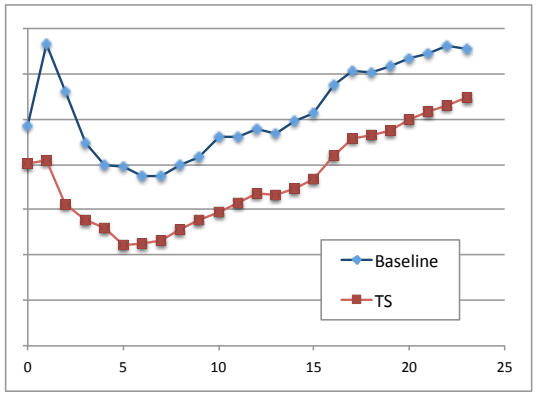

Figure 10: Evolution of eCPC on production traffic.

9. CONCLUSIONS

Multi-armed bandits have an important role to play in modern production systems that emphasize "continuous improvement", where products remain in a perpetual state of

feature testing even after they have been launched. In this paper we advocate using randomized probability matching as a superior algorithm, and the industry standard, due to its performance both in terms of accuracy, as well as speed, broad applicability and ease-of-use. Thompson Sampling offers superior real-time bid prediction compared to the baseline algorithm in just a few milliseconds.

10. REFERENCES

[1] S. Agrawal and N. Goyal. Analysis of thompson sampling for the multi-armed bandit problem. In *Proceedings of the 25th Annual Conference on Learning Theory (COLT)*, June 2012.

[2] S. Agrawal and N. Goyal. Further optimal regret bounds for thompson sampling. *CoRR*, abs/1209.3353, 2012.

[3] S. Agrawal and N. Goyal. Thompson sampling for contextual bandits with linear payoffs. *30th International Conference on Machine Learning (ICML)*, June 2013.

[4] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Mach. Learn.*, 47(2-3):235–256, 2002.

[5] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire. The nonstochastic multiarmed bandit problem. *SIAM J. Comput.*, 32(1):48–77, Jan. 2003.

[6] A. Blum. Empirical support for winnow and weighted-majority based algorithms: Results on a calendar scheduling domain. In A. Prieditis and S. J. Russell, editors, *ICML*, pages 64–72. Morgan Kaufmann, 1995.

[7] N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, New York, NY, USA, 2006.

[8] O. Chapelle and L. Li. An empirical evaluation of thompson sampling. In *NIPS*, pages 2249–2257, 2011.

[9] W. Ding, T. Qin, X.-D. Zhang, and T.-Y. Liu. Multi-armed bandit with budget constraint and variable costs. In M. desJardins and M. L. Littman, editors, *AAAI*. AAAI Press, 2013.

[10] B. Edelman, M. Ostrovsky, and M. Schwarz. Internet advertising and the generalized second price auction: Selling billions of dollars worth of keywords. Working Paper 11765, National Bureau of Economic Research, November 2005.

[11] Y. Freund, R. E. Schapire, Y. Singer, and M. K. Warmuth. Using and combining predictors that specialize. In F. T. Leighton and P. W. Shor, editors, *STOC*, pages 334–343. ACM, 1997.

[12] J. C. Gittins. Bandit processes and dynamic allocation indices. *Journal of the Royal Statistical Society. Series B (Methodological)*, 41(2):148–177, 1979.

[13] O. Granmo. Solving two-armed bernoulli bandit problems using a bayesian learning automaton. *International Journal of Intelligent Computing and Cybernetics*, 3(2):207–234, 2010.

[14] N. Gupta, O.-C. Granmo, and A. Agrawala. Thompson sampling for dynamic multi-armed bandits. In *Proceedings of the 2011 10th International Conference on Machine Learning and Applications and Workshops - Volume 01*, pages 484–489. IEEE Computer Society, 2011.

[15] E. Kaufmann, N. Korda, and R. Munos. Thompson sampling: An asymptotically optimal finite-time analysis. In *Proceedings of the 23rd International Conference on Algorithmic Learning Theory*, ALT'12, pages 199–213. Springer-Verlag, 2012.

[16] J. Langford. Tutorial on practical prediction theory for classification. *J. Mach. Learn. Res.*, 6:273–306, 2005.

[17] J. Langford, A. Strehl, and J. Wortman. Exploration scavenging. In *Proceedings of the 25th International Conference on Machine Learning*, pages 528–535, New York, NY, USA, 2008. ACM.

[18] K.-C. Lee, A. Jalali, and A. Dasdan. Real time bid optimization with smooth budget delivery in online advertising. In *Proceedings of the Seventh International Workshop on Data Mining for Online Advertising*, pages 1:1–1:9, New York, NY, USA, 2013. ACM.

[19] K.-c. Lee, B. Orten, A. Dasdan, and W. Li. Estimating conversion rate in display advertising from past erformance data. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '12, pages 768–776, 2012.

[20] L. Li. Generalized thompson sampling for contextual bandits. *CoRR*, 2013.

[21] L. Li, W. Chu, J. Langford, and R. E. Schapire. A contextual-bandit approach to personalized news article recommendation. In M. Rappa, P. Jones, J. Freire, and S. Chakrabarti, editors, *WWW*, pages 661–670. ACM, 2010.

[22] G. Marsaglia and W. W. Tsang. A simple method for generating gamma variables. *ACM Trans. Math. Softw.*, 26(3):363–372, Sept. 2000.

[23] M. Matsumoto and T. Nishimura. Mersenne twister: A 623-dimensionally equidistributed uniform pseudo-random number generator. *ACM Trans. Model. Comput. Simul.*, 8(1):3–30, Jan. 1998.

[24] B. C. May, N. Korda, A. Lee, and D. S. Leslie. Optimistic bayesian sampling in contextual-bandit problems. *J. Mach. Learn. Res.*, 13(1):2069–2106, June 2012.

[25] C. Perlich, B. Dalessandro, R. Hook, O. Stitelman, T. Raeder, and F. Provost. Bid optimizing and inventory scoring in targeted online advertising. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '12, pages 804–812. ACM, 2012.

[26] S. L. Scott. A modern bayesian look at the multi-armed bandit. *Appl. Stoch. Model. Bus. Ind.*, 26(6):639–658, Nov. 2010.

[27] W. R. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25:285–294, 1933.

[28] S. Jafarpour, V. Cevher, and R. E. Schapire. A game theoretic approach to expander-based compressive sensing. *Proceedings, IEEE International Symposium on Information Theory*, 2011.

[29] R. E. Schapire, and Y. Freund. Boosting: Foundations and Algorithms *MIT Press*, 2012.