Conversion-Based Dynamic-Creative-Optimization in Native Advertising

Yohay Kaplan, Alex Shtoff, Tomer Shadi, Oren Somekh
Yahoo Research, Haifa, Israel
{yohay,alex.shtoff,tomer.shadi,orens}@yahooinc.com

Yair Koren
Meta, Tel-Aviv, Israel
yairkoren@meta.com

Abstract—Yahoo Gemini native advertising marketplace serves billions of impressions daily, to hundreds of millions of unique users, and reaches a yearly revenue of hundreds of millions USDs. Powering Gemini native models for predicting advertise (ad) event probabilities, such as conversions and clicks, is OFFSET - a feature enhanced collaborative-filtering (CF) based event prediction algorithm. The predicted probabilities are then used in Gemini native auctions to determine which ads to present for every serving event (impression). Dynamic creative optimization (DCO) is a recent Gemini native product that was launched two years ago and is increasingly gaining more attention from advertisers. The DCO product enables advertisers to issue several assets per each native ad attribute, creating multiple combinations for each DCO ad. Since different combinations may appeal to different crowds, it may be beneficial to present certain combinations more frequently than others to maximize revenue while keeping advertisers and users satisfied. The initial DCO offer was to optimize click-through rates (CTR), however as the marketplace shifts more towards conversion based campaigns, advertisers also ask for a conversion based solution. To accommodate this request, we present a post-auction solution, where DCO ads’ combinations are favored according to their predicted conversion rate (CVR). The predictions are provided by an auxiliary OFFSET based combination CVR prediction model, and used to generate the combination distributions for DCO ad rendering during serving time. An online evaluation of this explore-exploit like solution, via online bucket A/B testing, serving Gemini native DCO traffic, showed a 53.5% CVR lift, when compared to a control bucket serving all combinations uniformly at random. The impressive results demonstrate the ability of this practical yet effective solution to overcome many real-life issues such as data sparsity, reporting delays, trends, and various system constraints. The CVR prediction based DCO system is now fully deployed, serving all Gemini native traffic.

I. INTRODUCTION

Yahoo Gemini native marketplace serves users with ads that are rendered to resemble the surrounding content. Operating with a yearly run-rate of several hundred million USDs, Gemini native is one of Yahoo main businesses. With more than two billion impressions daily, and an inventory of a several hundred thousand active ads at any given time, this marketplace performs real-time first price auctions that take into account budget considerations and targeting.

In order to rank the native ads for incoming users and their specific context according to the cost per click (CPC) price type where advertisers pay for clicks, a score (or expected revenue) is calculated by multiplying the predicted click-through rate (pCTR) by the bid for each active ad. The pCTR is provided by a model that is periodically updated by OFFSET - a feature-enhanced collaborative-filtering (CF) factorization-machines (FM) - like based event-prediction algorithm [1][2][3][5][9]. OFFSET is a one-pass (i.e., goes through the data only once) algorithm that updates its latent factor (LF) model for every new batch of logged data using an online gradient descent (OGD) based learning approach.

Dynamic creative optimization (DCO) is a recent product of the Gemini native portfolio Yahoo offers to its advertisers [11]. With DCO ads, advertisers may provide several assets for each native ad attribute, which are used to generate a plethora of combinations (see Fig. 1). Since different combinations may attract different crowds it may be beneficial to present certain combinations more frequently than others to maximize certain metrics such as click through-rate (CTR) and revenue. The DCO product provides a simple and quick way for advertisers to identify the best ads that may be assembled from groups of assets without conducting a time consuming and expensive in-house A/B testing.

As advertisers spend is shifting toward optimize CPC (oCPC) bidding strategy, where advertisers are interested in conversions1 but still pay for clicks, and since high CTR combinations do not necessarily entail high conversion rates (CVR) (i.e., the number of conversions divided by the number

1A conversion may follow (and be affiliated to) an impression (post-view) or a click (post-click), and may include events such as purchasing, registration, and app install.
of impressions), we were required to provide a conversion based DCO solution that favors high CVR combinations. However, the event counting based approach of [11] used for clicks is problematic when conversions are considered, since CVRs are generally two orders of magnitude lower than CTRs.

In this work we present the approach used to serve the DCO best combinations for increasing CVR in Gemini native marketplace. To maintain similar system resources (i.e., model sizes and serving latency), we adhere to a two-stage solution (see [4],[11]). During the first stage, the Serving system (or Serving) conducts a regular auction where all combinations of a DCO ads participate as a single native ad. The second stage is invoked only in case a DCO ad wins the auction, and a DCO combination is drawn according to some combination distribution (a *post-auction approach*). To calculate the combination distributions, we train an auxiliary combination CVR prediction model which is not consumed by the Serving system and is used to provide the combination predicted CVRs (pCVR) per certain traffic segments. The predictions are then turned into combination distributions where combinations having a higher pCVR are assigned with higher probabilities. As more conversions are accumulated, and certain DCO ad combinations are predicted to have higher CVRs, the distributions will drift from a uniform distribution, and the system will impress those combinations more frequently than others.

After a short alpha testing phase, optimizing internal DCO ads, the system was launched for a beta testing phase, serving traffic of selected advertisers. The conversion based DCO solution was compared to a control bucket serving all combinations uniformly at random, and showed a staggering 53.5% CVR lift. After the successful beta testing, the proposed conversion based DCO system is now fully deployed and available for all advertisers.

II. SERVING ADS TO USERS

When a user arrives at a Yahoo or third party partner site, and a native slot should be populated by an ad, an auction takes place. At first, the Serving system generates a list of scores for all eligible active ads. In general, an ad’s eligibility to a certain user in a certain context is determined by *targeting*, which is beyond the scope of this work, and relates to user characterization (such as age, gender, and geo) specified by the advertiser to approach certain crowds.

The score is a measure that attempts to rank ads according to their expected revenue with respect to the arriving user and her context (i.e., her features, e.g., age, gender, geo, day, hour, site, device type, etc.) and is roughly defined as

$$\text{Score}_{u,a} = \text{bid}_a \cdot \text{pCTR}_{u,a}$$

where \(\text{pCTR}_{u,a}\) (predicted click probability) is calculated by the Offset main click model (see [10, Eq. (1)])$^2$ for a click event, and \(\text{bid}_a\) (in USD) is the amount of money the advertiser is willing to spend for a click on a *manual cost per click* (mCPC) bidding strategy ad \(a\). For an *optimized cost per click* (oCPC) bidding strategy ad \(\text{bid}_a = \text{pCONV}_{u,a} \cdot \text{tCPA}\), where \(\text{pCONV}_{u,a}\) (predicted conversion probability given a click event) is calculated by the Offset main conversion model (see [10, Eq. (1)]) for a conversion-given-click event, and the *target cost per action* (tCPA) is the average cost the advertiser is expecting to spend on a conversion while still paying for clicks. Other bidding strategies such as *enhanced optimized cost per click* (eCPC), where part of the traffic is used to estimate the conversion cost, are outside of our scope. According to the *first price auction*, the winning ad is an eligible ad with the highest score \(\text{Score}_{u,a}\) (see Eq. (1)).

III. RELATED WORK

The problem of *dynamic creative optimization* (DCO), or finding the top combinations in terms of some metric such as CTR, CVR, or revenue, can be formulated as a *reinforcement learning* problem, where agents take actions to maximize some notion of cumulative reward [13]. In particular, our problem is closely related to the rich literature of the *multi-armed bandit* (MAB) problem, which comes in several flavors, such as *best arm identification* [6] and *regret minimization* [7]. Our problem is also closely related to the *online decision problem* [12]. Moreover, the use of an auxiliary prediction model to provoke actions is inspired by *Thompson sampling*, which is widely used to address such problems (see [8] for an online advertising application).

Our solution is based on a post-auction approach, which separates ad ranking and DCO ad rendering (as oppose to works that adopt a more holistic approach, e.g., [8]). The post-auction approach was used in [11] for a DCO product, however, with CTR as optimization goal. A *successive elimination* algorithm based on robust CTR measurements was used to identify the “best” combinations (see also [4] for a similar setting which considers *asset optimization* for Carousel ads). It is noted that although we consider a similar DCO product in the same native advertising marketplace as [11], we are interested in maximizing CVR which is much lower than CTR in general. To overcome the extreme data sparsity condition, we apply a totally different approach than that of [11], which is based on event prediction instead of event counting. In addition, the proposed solution does no “eliminate” combinations (as opposed to that of [11]) and inherently can follow combination CVR trends. It is worth mentioning that similar DCO products are offered also by other providers such as Facebook, Google, and LinkedIn.

IV. PROBLEM STATEMENT

Assume we have a DCO ad with \(M\) attributes and \(n_i\) assets for the \(i\)th attribute \(i = 1 \ldots M\). Therefore, there are \(N = \prod_{i=1}^{M} n_i\) combinations (or virtual native ad). The goal is to use the additional *degree-of-freedom* of having \(N\) combinations, to maximize the DCO ad CVR and increase revenue.

It is noted that by maximizing the CVR (as opposed to CTR maximization [11]) we cannot guarantee a direct increase in revenue since according to the oCPC bidding strategy, advertisers still pay for clicks. However, since our main click
and conversion-given-click prediction models are accurate, increasing the DCO ad CVR will cause the system to increase its bid (see oCPC explanation in Sec. II) pushing DCO ads to win more auctions and spend more (if budget allows). Moreover, for the mCPC bidding strategy DCO ads (see mCPC explanation in Sec. II), the advertisers are expected to increase their budgets since they pay less for their conversions if CVR increases. Therefore, in the long-run, CVR improvement is expected to increase revenue.

A. Requirements, constraints and conditions

Since the conversion DCO system should be incorporated into the existing Gemini native architecture, there are several system requirements and inherent constraints that considerably affect the nature of the selected solution.

Starting with the system requirements: (a) Model sizes – OFFSET click and conversion model sizes that are consumed by the Serving system cannot be considerably increased. Since most of the models consists of ad vectors, considering different combinations of DCO ads as new ads is not scalable; (b) Query-per-second (QPS) and service-level-agreement (SLA) – QPS cannot be considerably decreased (or alternatively SLA cannot be considerably increased). This means that the Serving system cannot make a query per DCO combination (e.g., using the combination number as an ad feature); (c) Serving complexity – computation complexity of the Serving system should be restricted to simple processing per ad (e.g., sigmoid function of the ads’ score according to [10, Eq. (1) and (2)]); and (d) for legacy reasons\(^3\) the interface to the Serving system is limited to combination distributions per DCO ad and certain traffic segments (see Sec. V-B). The Serving system then uses the combination distributions in a predefined manner to draw the DCO combination for rendering the DCO ad.

It is worth emphasizing that we are extremely sensitive to Serving system resources surge, as traffic is spread globally among many servers for maintaining harsh SLA requirements. However, we are almost agnostic to backend resources used to train our models, which is carried out over a couple of production grids.

Turning to the inherent constraints: (a) Delay – the time it takes for the system to update the model, index it, serve users, and log the users’ actions\(^4\). In the current production environment the average delay is more than an hour; (b) Incremental mode – data processing is currently done in batches of a few minutes to a few hours worth of data (e.g., 15 minutes and 4 hours for click and conversion models, respectively). This means that approaches that require policy changes after each action cannot be considered.

Finally, we describe the setting conditions that challenge the algorithm selection and system design: (a) Data sparsity – CVR is usually two orders of magnitude lower than CTR, hence, event counting based approach as the one used for optimizing click DCO ads [11] is problematic; (b) Reporting delays – conversion events may be reported up to 30 days after they occur. This is a real challenge that is hard to deal with since we do not want to wait so long before updating our models; (c) Varying impression rate – impression rates of DCO ads (as any other ad) are dictated mainly by their popularity, bid, and budget availability. These factors determine the ad delivery rate (the rate it wins auctions and gets impressions) and may change significantly over time; (d) Varying combination CVRs – CVR in general (as CTR) is varying over time (day over night, week days over weekends, etc.). In addition, popularity trends and ad fatigue may also cause ad (and combinations within DCO ads) CVR variations.

V. OUR APPROACH

A. Overview

Due to the constraints dictated by the Gemini native architecture (see Sec. IV-A) we chose to separate the processes of ad auction and DCO optimization, and adhere to a post-auction approach. Hence, for each incoming impression we let the Serving system use OFFSET models to select the winning ad (see Sec. II) and only then try to “optimize” the best combination for the current impression in case a DCO ad wins the auction. Since CVR is much lower than CTR, we use predictions instead of event counting to generate combination distributions. In particular, we train an auxiliary combination CVR prediction model (see Sec. V-D) and turn its predictions into combination distributions (see Sec. V-E) per DCO ad and certain traffic segments, defined by crossing contextual and demographic user features (see Sec. V-B) at the end of every training period. Then, the distributions (packed in a DCO model file) are periodically sent to the Serving system, allowing it to draw the selected combination according to the relevant distribution before rendering the winning DCO ad (see Sec. V-C). It is noted that we maximize CVR (probability of a conversion given an impression event) and not the probability of a conversion given a click event. We do that since after the auction, when the combination distributions are used by Serving, the impression has not happened yet and therefore CVR is the correct term to maximize.

We would like to emphasize that the novelty of our approach stems not from a novel conversion prediction modeling, but from our utilization of an existing well-proven conversion prediction framework to facilitate our light-weight post-auction approach to overcome challenges such as extreme data sparsity condition and harsh system requirements (see Sec. IV-A). For the benefits of our approach see [10, Sec. V-F].

B. Traffic segments definition

We define the traffic segments by configurable segment keys that relate to categorical unweighted user features (e.g., age, gender, and device). Assuming we use gender (i.e., male, female, and unknown) and device (i.e., mobile, desktop, tablet, and unknown) as segment keys, then for each DCO ad we have \(3 \times 4 = 12\) segments (e.g., female × desktop). For each DCO logged impression and conversion events, the segment keys are extracted and used as user features for training the
auxiliary prediction model (see Sec. V-D). During serving time, in case a DCO ad wins the auction, the segment keys are extracted from the incoming user’s features, and used to locate the corresponding traffic segment distribution for drawing the combination and rendering the DCO ad (see Section V-C).

C. Rendering DCO ads during serving time

Assume the auction winning ad is some eligible DCO ad \( A \) with \( N \) combinations \( C_A = \{C_n\}_{n=1}^N \). The Serving system extracts the segment keys of the incoming user’s features to determine the traffic segment \( S \) and locates the corresponding combination distribution \( Q_{A,S} = \{Q_{C_n}\}_{n=1}^N \) in the DCO model file. Then, it draws one combination according to \( Q_{A,S} \) before rendering \( A \).

D. Auxiliary combination CVR prediction model

To predict DCO ads combinations’ CVRs, we train an OFFSET based event prediction model (see [10, Sec. II-A]).

- Impressions are used as negative events while conversions are used as positive events.
- Since conversions are reported with long delays (up to 30 days after the actual view or click), we do not join the conversion and the impression that entailed it before training the model. Hence, each positive event is also trained as a negative event. Therefore, CVRs are slightly under-predicted and should be bias-corrected before turned into distributions (see Section V-E2).
- To reduce the backend resources required to train the auxiliary model we downsample the negative events (i.e., impressions) before training. Hence, predictions should be bias-corrected also to account for this operation before turned into distributions (see Section V-E2).

It is noted that unlike the main click and conversion-given-click models that are consumed (or indexed) by the Serving system for ranking the ads (see Section II), the auxiliary model is used only for generating DCO combination distributions. In addition, we train the auxiliary model over all conversion ad traffic and not only DCO ad traffic for triggering collaborative filtering patterns that help in “filling” the gaps due to conversion data sparsity. Next, we consider the ad and user features used to train the auxiliary model, and tie the latter to the traffic segments defined in Section V-B.

1) User features: Since the auxiliary model is used to predict the DCO ads combinations’ CVRs for each traffic segment of interest (see Section V-B), we use the corresponding segment keys as user features. For example, if segments’ traffic is defined by gender and device, then we use those as user features. It is noted that adding more user features may provide more accurate CVR predictions. However, there is little benefit in doing that in our setting, since we are interested in the CVR predictions of certain traffic segments that are defined by certain segment keys.

2) Ad features: We may use all “standard” ad features available for Gemini native models, such as \( ad \ ID \), \( advertiser \ ID \), and \( ad \ category \) (a multi-value feature with few dozens of values such as \( sports \) and \( electronics \) - see [10, Sec. II-A1]). To predict the CVR for each combination of each DCO ad, we require a special ad feature that ties together the assets that form each combination.

   a) Combination assets ad feature: Each event (impression or conversion) which involves a DCO ad, includes information regarding the actual assets used to render the ad. For example, assuming a certain DCO ad was impressed using a certain combination with description ID = De123, image ID = Im456, and title ID = Tt789, then a weighed multi-value feature \( \{\{De123,1,Im456,1,Tt789,1\}\} \) with unit weights (see [10, Sec. II-A1]) is generated and used to train the auxiliary model. It is noted that assets multi-value representation is used instead of representing the combination number as an ad feature, to create dependencies among the DCO ad’s combinations that share assets (e.g., share the same title and description but include different images). This in turn triggers collaborative patterns that help “filling” the gaps caused by data sparsity.

   b) Impressions and Conversions: For all events that involve non-DCO ads, the feature is assigned with “NONDCO” value and a unit weight, i.e., \( \{\{NONDCO,1\}\} \).

E. Turning predictions into distributions

At the end of every training period of the auxiliary prediction model (e.g., 4 hours), we scan it for DCO ads. Then, for each DCO ad we use the auxiliary model to calculate the DCO ad combinations’ CVR predictions for each traffic segment. To do that we gather the particular DCO ad features values (see Section V-D2), query the model, and construct all the DCO ad’s combinations latent-factor (LF) vectors (see [10, Sec. II-A1]). Since the system is designed such that traffic key segments coincides with the auxiliary prediction model user features, to get the user LF vector we query the model using the segment keys values that define the segment (e.g., female and desktop for gender and device segment keys), and construct the user vector (see [10, Sec. II-A]). After we have the user and combinations LF vectors, we extract the model bias and use [10, Eq. (1)] to calculate the combinations’ CVR predictions. Next, the predictions are bias-corrected to compensate for the non-join and impression downsampling operations (see Section V-E2). Finally, we use SoftMax\(^6\) to translate the true predictions into distributions and add a uniform component \( \lambda \) (for exploration purposes) to produce the final distribution. All distributions are gathered into a DCO distributions file and periodically sent to the Serving system to be used during serving time (see Section V-C). For a formal description of the predictions-to-distributions (P2D) algorithm see [10, Alg. 1].

1) Why SoftMax?: Our choice of using SoftMax facilitates a controlled mechanism to provide an explore-exploit trade-off and lets the system follow trends while presenting “better” combinations more frequently. However, there are other solutions that hold similar characteristics. In [10, Sec. V-E1] we provide a theoretical justification to our choice.

\(^6\)See https://en.wikipedia.org/wiki/Softmax_function.
2) Prediction correction: As mentioned earlier we down-sample the impressions (e.g., factor $r_{ds} = 100$) to reduce the auxiliary model training resources. In addition, we also do not “join” conversions with their impressions to avoid long training delays (conversions may be reported up to 30 days after they occur). Therefore, to get the “correct” CVR prediction we have to compensate for these two actions. The following is a rough approximation that is accurate only on average. Assuming we have $V$ conversions and $S$ skips (i.e., impressions without conversions) for a certain ad. Then, the average “raw” CVR with downsampling of $r_{ds}$ and non-join operation may be written as

$$\text{CVR}' = \frac{V}{V + \frac{V + S}{r_{ds}}}.$$  

Since the correct average $CVR = V/(V + S)$, we get

$$\text{CVR} = \min \left\{ 1, \frac{\text{CVR}'}{r_{ds}(1 - \text{CVR}')} \right\},$$  \hspace{1cm} (2)

where the minimum operation is required to keep $\text{CVR} < 1$ for very high average “raw” conversion rates $\text{CVR}' > r_{ds}/(1 + r_{ds})$ which is $\sim 0.99$ for $r_{ds} = 100$.

VI. PERFORMANCE EVALUATION

In this section we report the online performance of the conversion based DCO system. We describe the setting and baselines, define the performance metrics, present the results and discuss them.

a) Settings: After a phase of offline evaluation used to select the auxiliary model features and parameters, the conversion based DCO system was launched for an alpha online test phase, optimizing internal ads. The online tests demonstrated the potential of the proposed solution and were also used for tuning various system parameters. As a result, we trained the auxiliary model with all available ad features (see Section V-D2). As user features we use gender and device that are also used as traffic segment keys (a total of $3 \times 4 = 12$ segments). In addition, we set the user features independent and overlapping LF vector sizes to $s = o = 12$, so the final ad and user LF vector sizes are $D = 36$ (see [10, Sec. II-A]). It is noted that other parameters such as OGD step-size, AdaGrad parameters, and regularization parameter, are tuned automatically by OFFSET built-in online hyper parameter tuning mechanism (see [10, Sec. II-A] and [2]). For the P2D algorithm (see [10, Alg. 1]), we set the SoftMax factor $\beta = 13.86$ (so a 10% predicted CVR difference between the “best” and runner-up combinations results in a probability ratio of 4), and the uniform component $\lambda$ times the number of combinations to 0.1 (i.e., 10% of the probability mass). The aforementioned parameter set demonstrated the best CVR lifts we experienced in the limited grid search we performed during our online alpha test phase.

After demonstrating good online performance, the system was pushed to production (serving 90% of all traffic) for a beta test phase, done with a few selected advertisers, and included several dozens of conversion DCO ads with a total daily average of over million impressions. Although the beta experiments were coordinated, the advertisers controlled all aspects of their conversion DCO ads such as ad content, daily budget, and target CPA (tCPA).

b) Baselines: At any stage we compared the conversion based DCO bucket metrics to those of two control buckets each serving 5% of the traffic. The Uniform bucket was operating the CTR based DCO system with no conversion based DCO system, and rendering DCO ads using combinations that are chosen uniformly at random. The CTR bucket was operating the CTR based DCO system of [11] which is aimed at maximizing the DCO combinations’ CTR.

c) Metrics: To evaluate our system we use the following.

- CVR - conversion rate, i.e., the number of conversions divided by the number of impressions, where higher is better. We expect positive lifts if our system works properly.
- CTR - click through rate, i.e., the number of clicks divided by the number of impressions, where higher is better. We do not expect a major CTR lift over the Uniform baseline, since our system is not designed to maximize CTR. Moreover, we do expect a CTR drop in comparison to the CTR baseline which is designed to maximize CTR.
- Delivery - the number of impressions of the targeted traffic (i.e., DCO traffic).
- CPM - cost per thousand impressions, i.e., the total cost divided by the number of impressions and multiplied by 1000, where higher is better.
- CPA - cost per action, i.e., the total cost divide by the number of conversions, where lower is better. Lower CPA means advertisers spend less for each conversion.

While the CVR, CTR, and Delivery metrics, are easily measured and interpreted, the CPM and CPA metrics, as we shall explain in the sequel, should be handled more carefully.

d) Results and discussion: The results of the conversion based DCO system beta test are summarised in Table I. We emphasize that the reported results were measured over the traffic of all DCO ads (oCPC and mCPC) that had at-least one conversion during the evaluation period, while other DCO ads traffic is ignored. Over a period of 28 days (four weeks), we measured an impressive 53.5% and 38.0% average CVR lifts of the conversion based DCO bucket in comparison to the Uniform and CTR buckets, respectively. Since the performance were measured over millions of impressions and the lift is large, the superiority of our approach over the baselines is established with high confidence. In particular, the positive CVR lifts are statistically significant with $p-value < 10^{-14}$.

\begin{table}[h]
\centering
\begin{tabular}{|c|c|c|c|c|c|}
\hline
Baseline & CVR lift & CTR lift & Delivery lift & CPM lift & CPA lift \\
\hline
Uniform & 53.5\% & 1.6\% & 0.01\% & 0.44\% & -34.6\% \\
CTR & 38.0\% & -0.92\% & 0.00\% & -3.57\% & -30.1\% \\
\hline
\end{tabular}
\caption{Conversion based DCO test phase results summary.}
\end{table}

\footnotetext{Lifts are calculated by $(M_{DCO}/M_{baseline} - 1) \cdot 100$, for any positive metric $M > 0$.}
We also measured a 1.69% average CTR lift ($p-value = 0.016$) over the Uniform baseline and only a $-0.92\%$ CTR drop (negative lift) when compared to the CTR baseline, which means that there is some positive correlation between DCO combinations that entail more conversions and those that entail more clicks. However, it is quite evident that the system is indeed tuned for optimizing CVR. Moreover, the measured CTR drop in comparison to the CTR baseline is expected since the latter is designed to maximize CTR.

Examining the minuscule Delivery lifts of 0.01% and 0.00%, it is concluded that the targeted traffic of DCO ads with at least one conversion is almost equal (after proper normalization) between the conversion based DCO and baseline buckets. This is easily explained by recalling that both buckets rank the ads using the same main click and conversion-given-click prediction models which are trained over the entire traffic. Moreover, the DCO combination selection is done after the auction and has no direct influence on the delivery. It is also noted that since normalized delivery is almost equal between all buckets, CPM lift, is actually revenue lift.

The measured CPM lift of 0.44% over the Uniform bucket seems quite disappointing and does not reflect the staggering CVR lift. However, the fact that all buckets use the same main models for ad ranking and therefore have similar deliveries, combined with the modest CTR lift and the fact that all DCO ads pay for clicks, explains the CPM result. Moreover, the baselines actually “benefit” from the increased CVR of DCO ads caused by the conversion based DCO system. This is since the main ranking models are trained over the entire traffic which is dominated by the conversion based DCO system traffic (90% of all traffic). The measured $-3.57\%$ CPM significant drop when compared to the CTR baseline may be explained by similar reasons and by the fact that we actually measure also a CTR drop. It is concluded that in the current A/B test setting where the main ranking models are dominated by the conversion based DCO system traffic, we can not measure actual revenue lifts. As a matter of fact, the conversion based DCO system is expected to increase revenue proportionally to the CVR lift. This is since the expected revenue for oCPC ads, which consist of the lion share of the targeted DCO traffic, is $pCTR \cdot pCONV \cdot tCPA \approx pCVR \cdot tCPR \approx CVR \cdot tCPA$ (see Section II) assuming our main models are accurate, no CPA drop is expected for our main models are accurate, no CPA drop is expected for our main models are accurate, no CPA drop is expected for our main models are accurate, no CPA drop is expected for our main models are accurate, no CPA drop is expected for our main models are accurate, no CPA drop is expected for our main models are accurate, no CPA drop is expected for our main models are accurate, no CPA drop is expected for oCPC DCO ads. It is noted that the evaluation was dominated by a few active DCO ads with relatively high CVRs. Milder CVR lifts are expected for an evaluation done with larger and more typical DCO demand.

VII. CONCLUDING REMARKS

In this work we presented a simple yet practical post-auction two-stage solution to the conversion optimization DCO problem. During the first stage of serving, a regular Gemini native auction is performed and if a DCO ad wins the auction, the Serving system uses the combination distributions generated by our combination CVR prediction based DCO solution, to draw the rendered combination. Since the solution is based on predictions provided by an auxiliary prediction model, and not on event counting used by the CTR based DCO system of [11], it is capable of following trends and mitigate data sparsity issues. After showing good online performance, significantly increasing the DCO ads’ CVRs, the conversion DCO system was deployed and has been serving all Gemini native traffic since. We note that different approaches such as (a) consider each DCO combination as a new native ad; and (b) treating the different combinations as ad features, were considered and were not perused due to their excessive resources requirements. Future work may include improving the auxiliary prediction model accuracy and combine CTR/CVR based prediction models to provide a unified DCO solution.

REFERENCES

[1] M. Aharon, N. Aizenberg, E. Bortnikov, R. Lempel, R. Adadi, T. Benyamini, L. Levin, R. Roth, and O. Serfaty. Offset: one-pass factorization of feature sets for online recommendation in persistent cold start settings. In Proc. RecSys’2013.
[2] M. Aharon, A. Kagian, and O. Somekh. Adaptive online hyperparameters tuning for ad event-prediction models. In Proc. WWW’2017.
[3] M. Aharon, Y. Kaplan, R. Levy, O. Somekh, A. Blanc, N. Eshel, A. Shahr, A. Singer, and A. Zlotnik. Soft frequency capping for improved ad click prediction in yahoo Gemini native. In Proc. CIKM’2019.
[4] M. Aharon, O. Somekh, A. Shahr, A. Singer, B. Trayvas, H. Vogel, and D. Dobrev. Carousel ads optimization in yahoo Gemini native. In Proc. KDD’2019.
[5] M. Arian, E. Abutbul, M. Aharon, Y. Koren, O. Somekh, and R. Stram. Feature enhancement via user similarities networks for improved click prediction in yahoo Gemini native. In Proc. CIKM’2019.
[6] J.Y. Audibert and S. Bubeck. Best arm identification in multi-armed bandits. In Proc. COLT’2010.
[7] S. Bubeck, N. Cesa-Bianchi, et al. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. Foundations and Trends® in Machine Learning, 5(1):1–122, 2012.
[8] O. Chapelle and L. Li. An empirical evaluation of Thompson sampling. In Proc. NIPS’2011.
[9] Y. Kaplan, Y. Koren, R. Leibovits, and O. Somekh. Dynamic Length Factorization Machines for CTR Prediction In Proc. IeeeBigData’2021.
[10] Y. Kaplan, Y. Koren, T. Shadi, and O. Somekh. Conversion-Based Dynamic-Creative-Optimization in Native Advertising arXiv, 2022.
[11] Y. Koren, O. Somekh, A. Shahr, A. Izhaki, T. Cohen, M. Krasteva, and T. Shadi. Dynamic creative optimization in Verizon media native advertising. In Proc. IeeeBigData’2020.
[12] D.J. Russo, B. Roy, A. Kazerouni, T. Oshband, and Z. Wen. A tutorial on Thompson sampling. Foundations and Trends® in Machine Learning, 11(1):1–96, 2018.
[13] R.S. Sutton and A.G. Barto. Reinforcement learning: An introduction. MIT press, 2018.