Abstract

MODELING AND EVALUATION FRAMEWORK FOR RESPONSIVE SEARCH ADS

Jiamei Shuai, Microsoft Advertising, jiashua@microsoft.com
Shuayb Zarar, Microsoft Advertising, shuayb@microsoft.com
Denis Charles, Microsoft Advertising, cdx@microsoft.com

Keywords. Reinforcement learning, sequential contextual bandits, exploitation and exploration, Thompson sampling, uncertainty estimation, logistic regression, deep neural networks.

Search-ads is an important product in computational advertising. In order to achieve high performance in search-ads, it is critical to improve ad-copy relevance and click through rates. Traditionally, the onus has been on advertisers to come up with an ad-copy and set up appropriate bids that can achieve these challenging objectives. Thanks to the emergence of AI and machine-learning technologies, we can simplify both the bidding and ad-copy generation processes. Responsive search ads (RSA) is an emerging product in this direction that tackles the ad-copy generation piece. RSA provides advertisers with sufficient control over the ad-copies that can be shown for their products by giving them the option of providing components (title and description assets) of the ad-copy, as opposed to the ad-copy itself, and tackling the assembly of these components dynamically at runtime. This allows us to take into consideration query and user contextual information and produce better ads. Even with a small number of assets, there are many potential asset combinations. In this work, we propose a large-scale logistic-regression model, combined with a sequential contextual-bandits framework that allows us to assemble high-performing ad-copies at runtime. We have productized our algorithms as part of Bing Ads. The shipped models have achieved 6% adoption based on servable customer count and have already shown increasing daily revenue as well as promising marketplace KPI trends of all up Revenue-per-Mille (RPM), Click-Yield (CY) and Impression-Yield (IY) in just a few months' time. In this paper, we will discuss the modeling details, evaluation techniques as well as online flight metrics with real search traffic from bing.com.

1. Introduction

In digital advertising, a text ad (TA) that shows up along with search results, also known as a search ad, is composed of titles, descriptions, typically referred to as assets, and other pieces of information. An anatomy of a search ad from bing.com is shown in Figure 1. It is interesting to observe that not only the ad-copy is intelligible but also the individual assets – titles and descriptions – make sense when viewed in isolation **(Hillard et al., 2010; Shaparenko et al., 2009)**.

Responsive Search Ads (RSA) is an emerging ad product that takes advantage of the above observation to simplify the ad-creation and testing process. It requires advertisers to provide individual title and description assets, as opposed to providing the entire ad-copy. At runtime, RSA exploits machine learning to optimize each ad by mixing and matching the headlines and descriptions to produce different ad-copies for the same search ad.

Figure 2 shows an overview of the RSA product. Advertisers provide 3 - 15 titles and 2 - 4 descriptions as part of the ad-creation process. Given that each RSA ad contains 3 title positions and 2 description positions, these pieces can be combined in approximately 32,000 ways for the maximum case. Each combination of an RSA ad has specific relevance to a query and likelihood of being clicked by a user. RSA utilizes a machine-learning model to select a combination that can maximize both metrics at runtime.

In this paper, we present details of this machine-learning model along with offline and online results. Specifically, we utilize Logistic Regression (LR) for predictive modeling and augment it with an exploration mechanism based on the sequential contextual-bandits framework **(Beygelzimer et al., 2011; Dudík et al., 2011; Bietti et al., 2010; Zhang et al., 2019)**. We have shipped this model to general availability (GA) in January 2020, and it currently serves live traffic from bing.com in USA and several international markets.

Figure 3 shows the increase in adoption of the RSA product in terms of customer volume, revenue, impressions, clicks and conversions. This is an important product for bing.com and we continue to see increased adoption across all geographies and advertiser groups.

2. RSA Modeling

As mentioned before, the problem that we tackle is to simultaneously estimate the relevance of an ad-combination to a query and its likelihood of being clicked by a user at runtime. In this section, we present details of the objective function that we utilize followed by the prediction and exploration models.

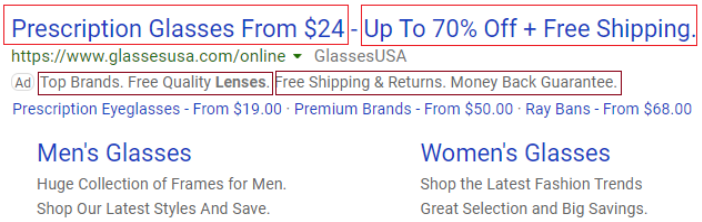


Figure 1: A search ad a.k.a. text ad comprises titles and descriptions (shown as boxes), sitelinks and other decorations.

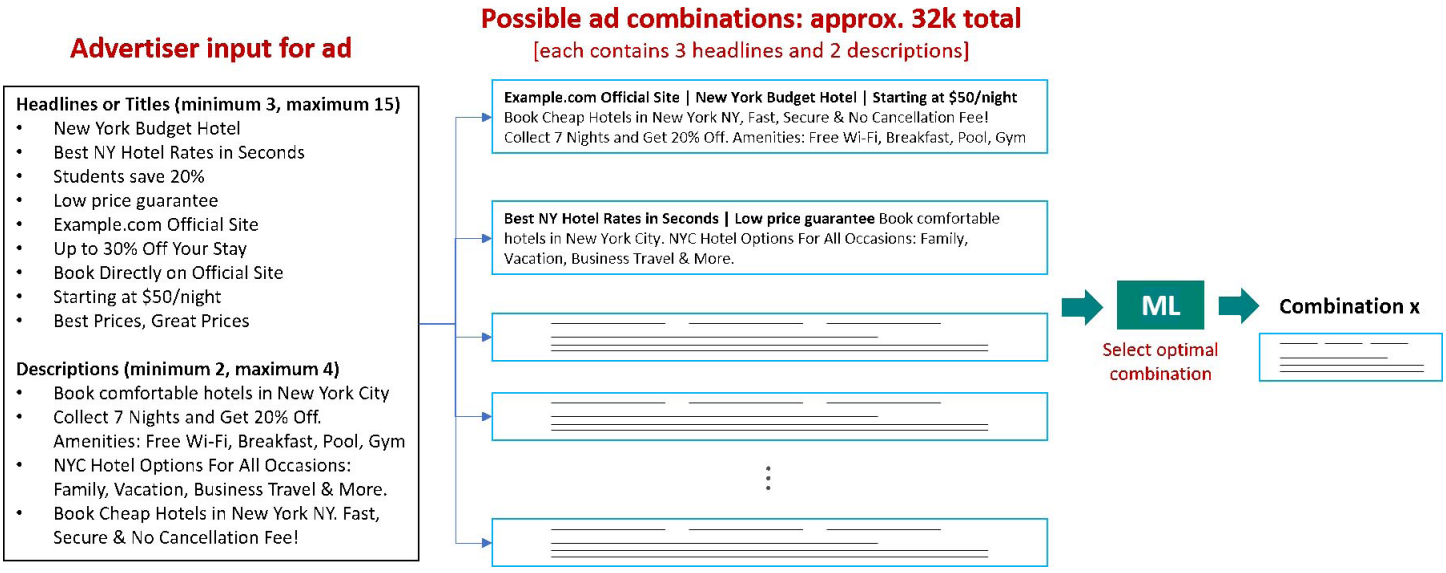


Figure 2: RSA simplifies ad-creation process. It requires advertisers to provide headline (up to 15) and description (up to 4) assets. It utilizes machine-learning to select assets and stitch together an ad-copy depending on the user and query context at runtime.

2.1. Objective Function

In online ads serving, the generated combinations of RSA will be passed to the downstream stage of auction. Only ads with relatively better quality and click through rate could win the process and thus not be filtered. To model the RSA asset stitching as a supervised learning problem, we approximate the two RSA goals (relevance of ad-combination to query and its likelihood of being clicked by a user) through winrate, which is defined per asset combination as follows:

$$\text{winrate}(c_i) = \frac{\text{number of times } c_i \text{ wins in auction}}{\text{number of times } c_i \text{ participates in auction}}$$

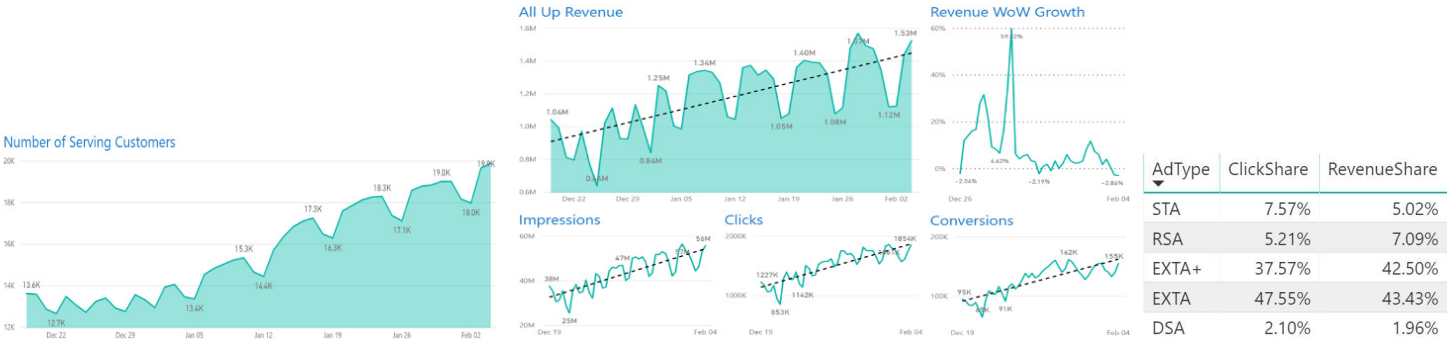
We denote the specific asset combination we are evaluating as c_i. Winrate measures how often an ad-combination wins an auction. It implicitly captures the required relevance and click metrics for RSA ads. This is because to win an auction, we select asset combinations that have a high Ad Rank (also known as Rank Score), which is computed as follows:

$$\text{Ad Rank} = \text{CPC Bid x Quality Score}$$

Cost-per-click (CPC) bid depends on extraneous factors such as advertiser budgets. However, quality score is based on the expected click through rate (likelihood that the asset combination will be clicked) and ad relevance (how closely the asset combination matches the intent behind a user's search query). Thus, a high ad rank, and in turn a high win rate, is a reasonable proxy for the RSA goals. Another reason we select winrate as the objective is because it is easily measurable in our online system.

2.2. Prediction Model

The key challenge we have in the prediction problem is that at training time we have labels per asset combination *i.e.*, whether an asset combination won or lost the auction. We can indeed train a supervised learning model with these labels per asset combination. However, at the time of online inference, we are not able to score all possible combinations that can be derived from a set of assets. This is because, we have many combinations, specifically 15×14×13×5×4 = 32,760 in the maximum case. Therefore, we rely on domain knowledge that the initial asset positions have a greater influence over the win or loss outcome than the later positions in the asset combination. Therefore, at scoring time, we utilize a subsection of the supervised model to sequentially select assets per position starting from the first position. In the of this section, we present more details about this model.



AdType	ClickShare	RevenueShare
STA	7.57%	5.02%
RSA	5.21%	7.09%
EXTA+	37.57%	42.50%
EXTA	47.55%	43.43%
DSA	2.10%	1.96%

Figure 3: RSA has achieved 6% adoption based on servable customer count and increasing trends in revenue, impression volume, clicks, conversions and share with respect to other existing ad products such as STA (search text ads), EXTA (extended text ads) and DSA (dynamic search ads).

• ASSET LENGTH=4
•ASSET UNIBIGRAM:
1.H(YOUR)
2.H(PERFECT)
3.H(USED)
4.H(BMW)
5.H(YOUR PERFECT)
6.H(PERFECT USED)
7.H(USED BMW)
•ASSETHASH:
H(YOUR PERFECT USED BMW)

• TERM PAIRS(QUERY,TITLE2)
1.H(USED)_X_H(YOUR)
2.H(USED)_X_H(PERFECT)
3.H(USED)_X_H(USED)
4.H(USED)_X_H(BMW)
5.H(BMW)_X_H(YOUR)
6.H(BMW)_X_H(PERFECT)
7.H(BMW)_X_H(USED)
8.H(BMW)_X_H(BMW)
9.H(SALE)_X_H(YOUR)
10.H(SALE)_X_H(PERFECT)
11.H(SALE)_X_H(USED)
12.H(SALE)_X_H(BMW)

FEATURETYPE	FEATURECOUNT
ASSETLENGTH	12
ASSETUNIBIGRAM	2,293,786
ASSETHASH	2,270,935
QHASHXMHASH	744,835,090
QTERMHASHXMHASH	239,728,387
QHASHXMTERMHASH	2,808,982,178
QTERMHASHXMTERMHASH	333,747,210
QLENXMLEN	124
QMATCHEDTERMXMHASH	3,212,554
QLENXMATCHEDLEN	116
MLENXMATCHEDLEN	106
QUNMATCHEDTERM	18,246,665
MUNMATCHEDTERM	193,890
QMATCHEDXQUNMATCHEDTERM	29,209,946
MMATCHEDXMUNMATCHEDTERM	2,299,811
QMATCHEDTERMXQMATCHEDTERM	1,058,988
QUNMATCHEDXMUNMATCHTERM	320,807,043

Tabke 1: On the left is an illustration of certain feature vector component for title 2. On the right is the full feature vector dimensionality, broken down by components for title 2.

Feature design. We utilize several features available in our system that capture structural details and textual content of the asset combinations. Specifically, we use string length, unigrams and bigrams from the asset text. We also cross these with the query text to produce a total sparse feature dimensionality per position of approximately 4B. Consider the search results shown below for a query "used bmw sale": take title 2 (*i.e.*, "Your Perfect Used BMW") as an example. The following is an example list of feature vectors that we extract. The full dimensionality of the feature vector components is also shown in Table 1 below on the right. At training time, we further cross the full feature vector set with position, add a global bias value and train an LR model as shown at the left in Figure 4. Due to business logic constraints, we are restricted to stitching an asset-combination that comprises five positions, namely three titles and two descriptions. Suppose X represents the combined feature vector from all five positions, i.e., $X = [X_1, X_2, X_3, X_4, X_5]$, and Y (equal to 0 or 1) represents the logged win or loss outcome for that specific asset combination. The LR model predicts the following probability:

$$P(Y = 1|X; W) = f(X, W) = \sigma(W^T X + b)$$

where W is the 5×4 billion dimensional weight vector that is learnt to minimize the binary cross-entropy loss $J(w)$ over M training examples as follows:

$$J(W) = -\frac{1}{M}\sum_{i=0}^{M} Y^{(i)} \log P(Y = 1) + (1 - Y^{(i)}) \log P(Y = 0)$$

At inference time, we have up to 32,760 combinations of assets that need to be scored with the LR model. However, as mentioned before, this is not feasible to be evaluated in real time. To fit our system-level constraints, we make some simplifying assumptions. It is a known fact that the first few positions in the title have more influence over the relevance and click-through rate metrics of an asset combination. Therefore, we perform a sequential per-position asset selection as shown at the right in Figure 4.

We utilize part of the trained LR model to score all available assets for position 1 (using weights W_1). Based on the predicted scores, we select the asset that achieves the highest winrate, after the exploration process described in the next section, for this position – shown as Asset 2 in Figure 4. We exclude this asset from the asset list and continue the scoring process for subsequent positions. Thus, we perform a greedy selection according to the following:

$$P(Y = 1|X_i; W_i) = f(X_i, W_i, b) = \sigma(W_i^T X_i + b), i \in [1, 5]$$

Thus, by breaking down the LR model, we can select assets by scoring only 49 assets per-user-query online – 15 assets in position 1, 14 in 2, 13 in 3, 4 in 4 and 3 in 5. The first three are titles and last two are descriptions.

2.3. Exploration model

We train the logistic regression model each day with new data that is collected as a result of RSA asset combinations being shown to users based on the previous days model. To provide low-performing assets a chance to be shown to users (these cases may also have a high winrate), and thereby debias our future training data by generating combinations that are likely to lose in the auction, we include a mechanism for exploration. This mechanism also helps us overcome some of the approximations that we have made at inference time to predict the online win or loss outcome by sequential per-position asset scoring as opposed to scoring at the level of asset combinations.

If we were to score all asset combinations there are 32,760 possible combinations, which are not only computational expensive to score by LR but also lead to a large action space

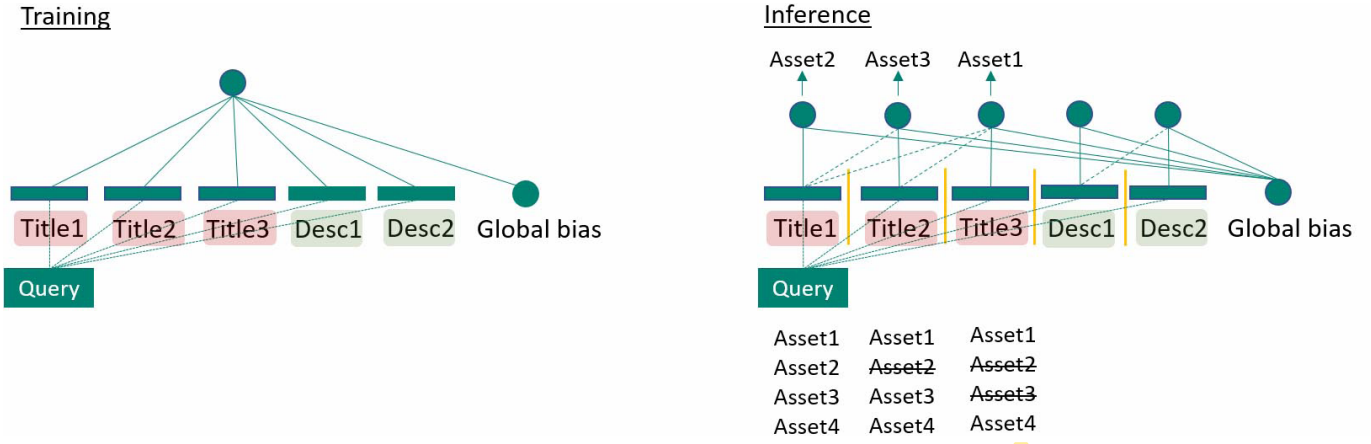


Figure 4: At training time, we cross the feature vectors with position, add a global bias and train a LR model with winrate as the objective. At inference time, we perform a sequential greedy asset selection starting from the first position.

for the exploration model. Unless we have a very large amount of data, it is well-known that common contextual bandit-based exploration algorithms do not perform well when they have a huge action space **(Abernethy et al., 2013)**. There are related approaches that solve similar problems through set-level estimation **(Zaheer et al., 2017)**. However, since we break down our prediction problem into per-position scoring, we can formulate exploration as a sequential contextual-bandits (CB) problem. The sequential CB exploration framework of RSA is shown in Figure 5. Specifically, we maintain five sets of CB policies or models, one per each asset position. The CB policies consume the LR scores and the features for each asset position to select an asset for each position. As mentioned before, since we do a sequential selection of assets, the action space for the policies, also denoted by arms in Figure 5, gradually reduces from 15 in position 1 to 14 in position 2 and so forth.

There are many options to choose for the CB policy. Typically, such problems are best solved through ϵ-greedy methods, where we randomly select an action in each step with some small probability, upper confidence bounds (UCB) **(Li et al., 2010; Chu et al., 2011)** where we employ some heuristics to estimate a confidence bound for the reward function so that the true value of the estimated reward using past experiences is within that bound with high probability, or Thompson Sampling where we select an action according to the probability that the action is optimal according to a specific underlying model **(Agrawal and Goyal, 2017; Chapelle and Li, 2011)**. Thompson Sampling is a randomized policy and has been shown to have near-optimal regret **(Agrawal and Goyal, 2017)**. Thus, we utilize it as the exploration model for RSA.

In our case, the action per-query per-position is the selection or rejection of a particular asset. The reward function is the outcome of the auction, which is a binary random variable. Thus, it follows a Bernoulli distribution. Suppose the probabilities of winning and losing the auction are denoted by p and $(1-p)$, respectively. Further assume the winning outcome p follows a beta-distribution, which is a conjugate prior of the Bernoulli distribution. Thus, we have:

$$P(p) = Beta(\alpha, \beta), \alpha = Np \text{ and } \beta = N(1-p)$$

α and β are parameters of the Beta distribution that allow us to control the level of exploration and exploitation during the decision-making process. These are dynamic parameters that are updated when we see the true outcome (auction win or loss) after performing the auction online. N is the number of observations which is the sum of all auction wins plus losses that have been made for the reward function at the time of

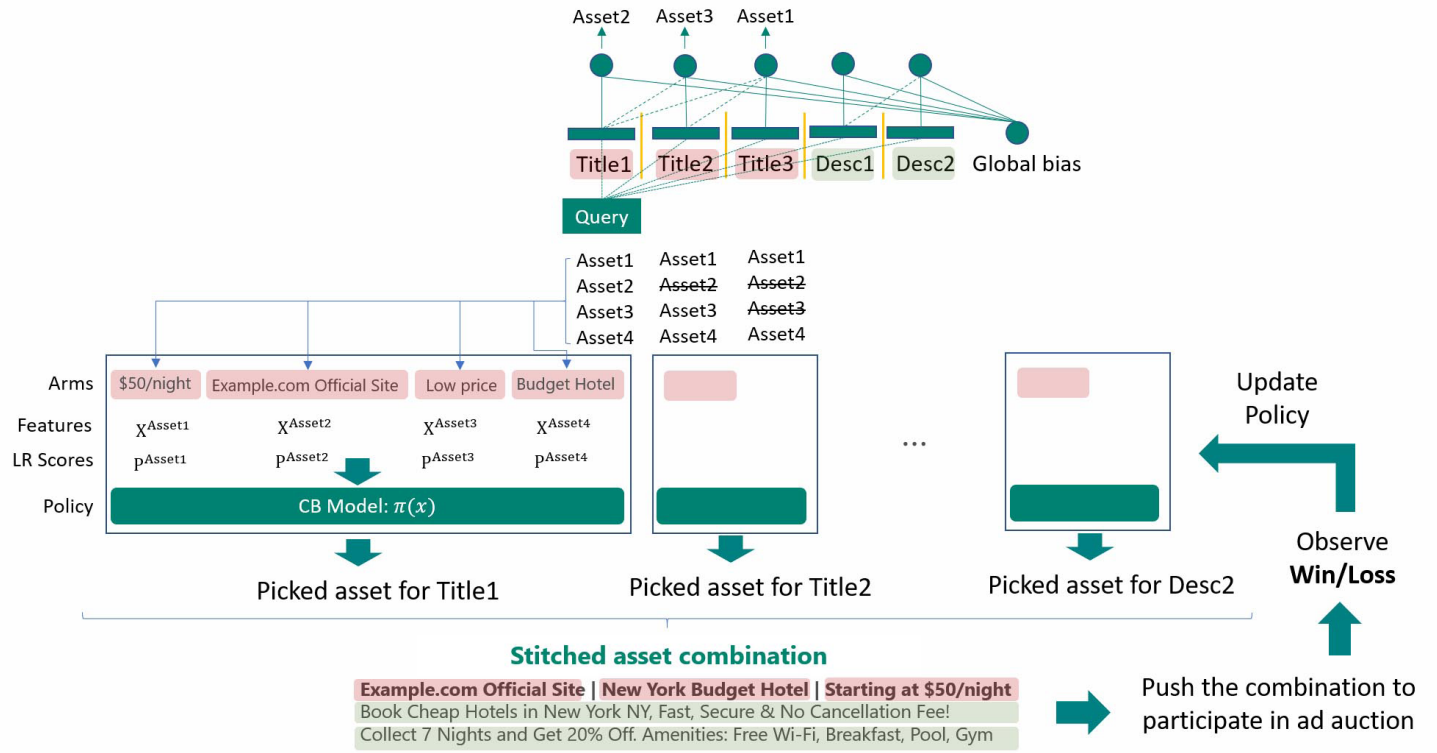


Figure 5: We select asset combinations after exploration with sequential contextual bandits. The CB model consumes the LR scores and features from each asset to select the one that has the most likelihood of success.

the sequential decision-making process. Smaller values of α, β and N lead to more exploration and larger values lead to more exploitation of the current logistic regression model as illustrated in Figure 6.

In our case, as we're doing a sequential per-position scoring, recall that the maximum asset numbers we need to score for each of the five positions are: 15, 14, 13, 5 and 4 respectively, in total 49. Thus we maintain in total 49 Beta distributions across all five positions as proxies for the posterior distribution of win-loss probabilities for each asset. For the first position, we sample 15 Beta distributions to estimate the probability of auction win for 15 assets, and then pick the asset that has the highest posterior probability value. We remove this particular asset from further evaluation. We move on to the second position and sample 14 beta distributions for the 14 assets, pick the asset that has the highest posterior probability value and then move on to the third, fourth and fifth positions for a similar evaluation. The sequential exploration process is illustrated in Figure 7.

In order to have the posterior probability depend on the asset text, we would like the Beta parameters to be computed from LR score and features that are available per asset. α and β already depend on the LR score (predicted win or loss probability). However, we need to come up with a good heuristic for the trial count N. For this, we utilize the gradient sum of the triggered features from the LR model and scale it with an adjustment factor (ratio).

$$N = \frac{\text{AdjustmentRatio}}{f(\text{gradient sum from model})}$$

In other words, as the LR model sees more and more data, its gradient sum becomes larger. $f(\cdot)$ is inverse proportional to the gradient sum. Therefore, the value of N gets larger. Thus, the model becomes more confident and reduces its exploration. The adjustment factor is a hyperparameter that helps keep the value of N within reasonable bounds.

3. Offline Experimental Evaluation

In this section, we present experimental results that validate our methodology on offline logs. In the next section, we present results from online flighting of the model; flighting refers to shipping the model and assigning live search traffic to it. An overview of the experimental framework is shown in Figure 9.

We utilize historical logs to train the LR model. We create a new log called IndexP log, which records all the asset combinations that participate in the auction; earlier, the existing IndexP log stored only those asset combinations that won the auction which ran in Listing Server and the

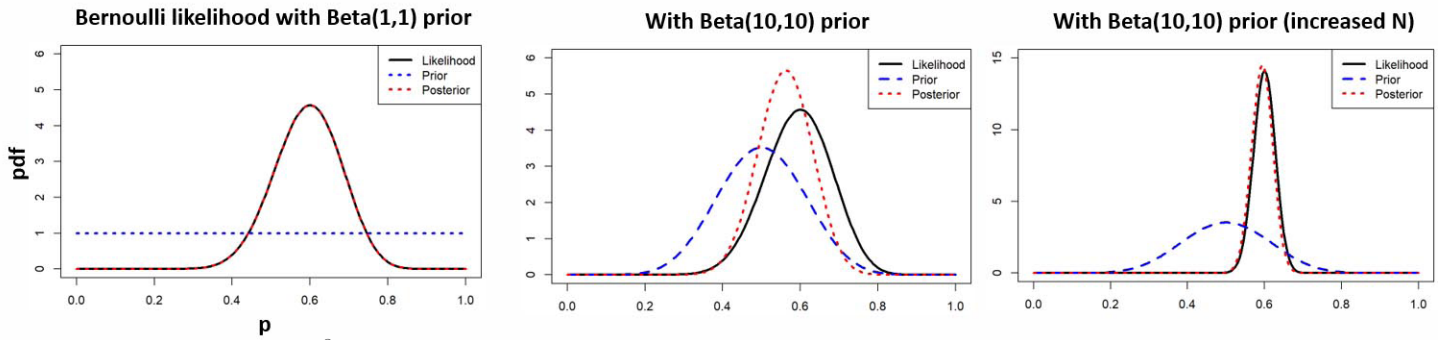


Figure 6: Smaller values of α, β and N lead to more exploration – wider range of the sampling probability on x-axis.

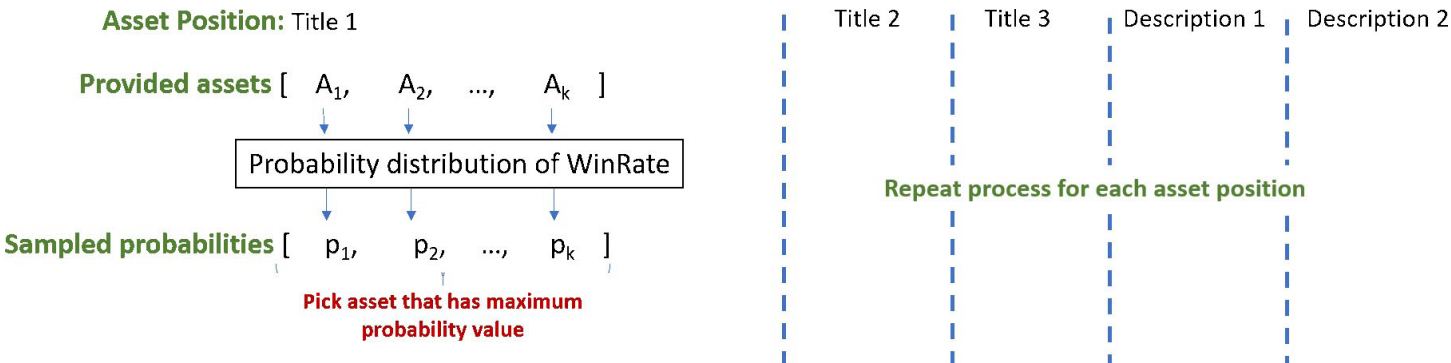


Figure 7: Samples are drawn from 15 Beta distributions in the first position and then the asset corresponding to the highest Beta score is selected for that position. Evaluation continues in a sequential manner across other positions.

hightest score lose one. The outcome of Listing Server is from a compute engine called the Delivery Engine (DE), which includes other components besides the ads auction. Furthermore, we create a new service in Creative Store called Caique. This service is where the RSA LR and exploration models run online.

For offline evaluation, the metrics we use are the loss of the LR model and Area under the curve (AUC) of the receiver operating characteristics (ROC) which are common performance measurements for classification problems **(Yi et al., 2013)**. However, since in the RSA ad-stitching task, the ranking process is among assets within an asset combination, as an approximation, we compute additional metrics such as per-advertiser winrate. Furthermore, in order to get an estimation of winrate offline for any new models, we employ an offline simulation based IPS scoring method.

Offline winrate simulation. When evaluating model's winrate through its own online traffic, different models have selection bias because of different distributions when selecting the samples. In offline evaluation, our target is to evaluate a new policy (model) using another existing policy's samples and labels of win or loss. A common strategy to remove this bias is propensity score weighting **(Dudík et al., 2011; Chan et al., 2010)**. Basically, we re-weight ad samples with weights inversely proportional to the probability of selection. The resulting score is also known as inverse propensity score (IPS). Suppose we have logged data of some base policy π_0 for each ad Ad_i, which consists of up to 15 title assets and 4 description assets, in our system in the form of $D_i = (x_1, a_1, p_1, r_1), \ldots, (x_n, a_n, p_n, r_n)$, where a_i is the asset combination from various asset combination options in this ad for which an action is logged in the system; x_i is the features corresponding to asset combination a_i; p_i is the probability that the asset combination a_i is selected by the base policy π_0 and r_i is the reward obtained when this asset combination was produced, which is auction win or loss in our case. In order to estimate the winrate of ad Ad_i under the assumption that it is processed by some new policy π_w instead of the logged base policy π_0, we can compute the IPS score as follows:

$$IPS_{Ad_i} = \sum_{j=1}^{n} \left[\frac{P(\pi_w(x_j) = a_j)}{p_j} r_j \right]$$

where $\pi_w(x_j)$ is the action with maximum predicted winrate among all asset combinations of the Ad_i under new policy π_w, and $P(\pi_w(x_j) = a_j)$ is the probability that asset combination a_j had that maximum value. $P(\pi_w(x_j) = a_j)/p_j$ is the IPS re-weighting factor.

To evaluate the efficacy of the CB framework, we utilize a randomized exploration scheme (referred to as Phase 1)

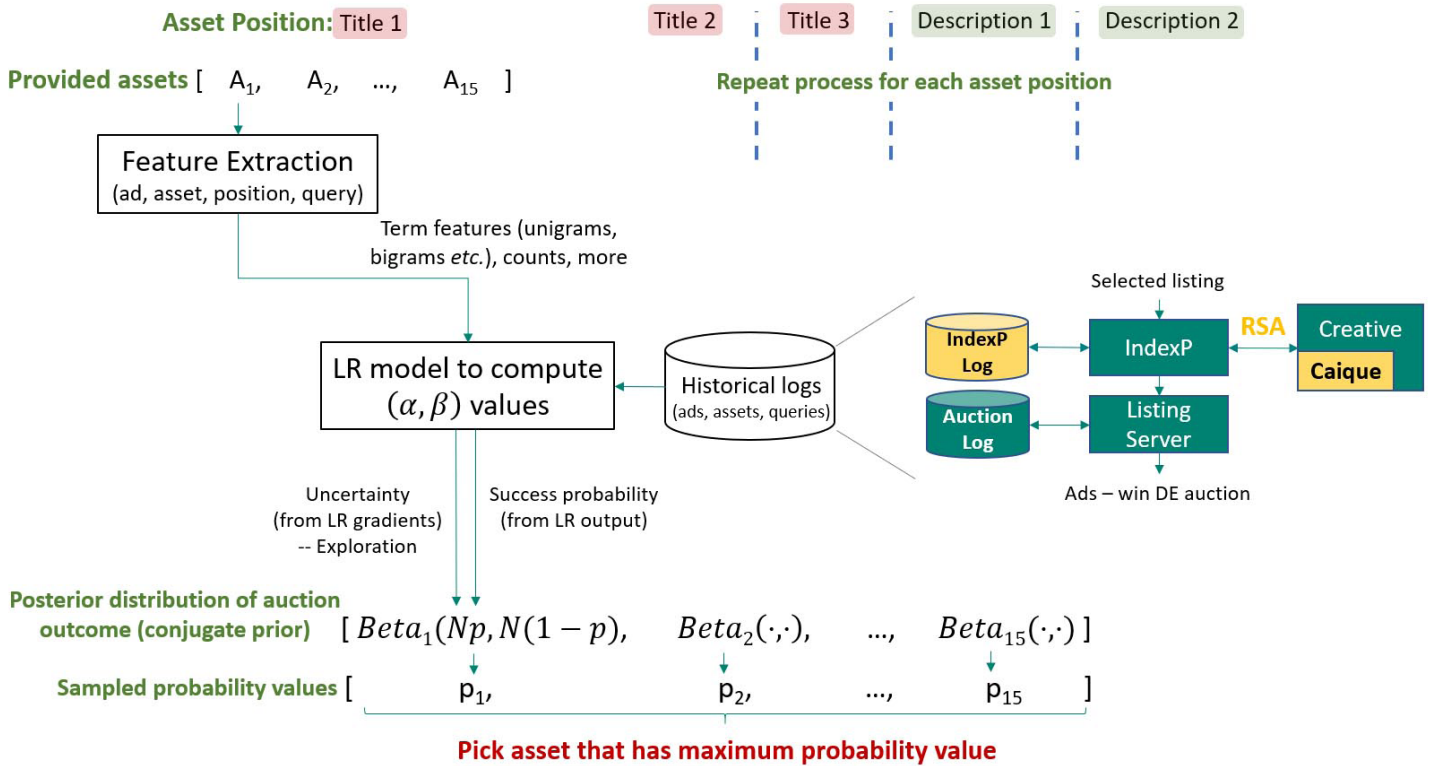


Figure 9: Our experimental framework relies on historical logs to train the LR model. We utilize the LR model to estimate the parameters of the Beta Distributions used in Thompson Sampling.

as well as a lookup-based approach, where we utilize historical logs directly to estimate the parameters of the Beta distribution *i.e.*, no LR model (referred to as Phase 1.5). We compare these approaches to the proposed approach (referred to as Phase 2). There are several asset combinations that are possible for each of these approaches. And, as shown in Figure 8, we pick combinations that intersect and evaluate the ad-level winrate using the IPS estimates above for each of these policies. As observed from the results in Table 2, we see that the proposed LR+SCB (logistic regression + sequential contextual bandits) Phase 2 approach achieves the highest per-ad win rate across all policies.

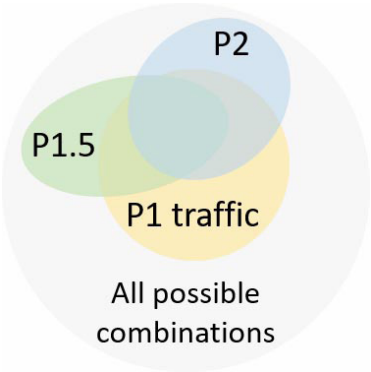


Figure 8: We use intersecting combinations across all.

Model	ListingAD	WinRate	Delta	StdDEV
PHASE 1	103,071	8.30%		
PHASE 1.5	103,071	8.48%	2.13%	0.072%
PHASE 2	103,071	8.60%	3.59%	0.074%

Table 2: LR+SCB model achieves the highest per ad winrate across combinations of 100k ads that we evaluated. The delta compared to a random exploration policy is 3.59%.

3.1. Prediction model results

In this section, we discuss impact of different hyperparameters on the modeling objective. Figure 10 shows that contextual features, which are currently captured by query-level information and by user-level information (future work), have high prediction power for winrate. More specifically, the LR+SCB model itself (v1) brings 26% AUC lift compared to the case where we do lookups based on historical logs (Phase 1.5). Furthermore, adding query features (v2) brings an additional AUC improvement of 11%. We observed that tuning L_1 parameters in FTRL **(McMahan et al., 2013)** leads to a smaller model size with little impact on the AUC. For instance, adding L_1=2 and 3 for title and description, respectively, leads to a validation AUC drop of about 0.36% but leads to almost 80% of the weights to be zero. This sparse model can be stored and served efficiently. We also tweaked the learning rate to 0.03 for FTRL. We note that one epoch of training on our daily data takes about 7 hrs. Training a second epoch takes the same amount of time but provides only 0.08% AUC lift. Therefore, we stick with a single pass over the data (single epoch) for training.

However, we continue to train the model each day by adding new demand – new RSA data from that day – to the training data. Figure 10 (b) shows the AUC drop when we use the same model over multiple days. And Figure 10 (c) shows the change in demand over multiple days.

4. Online Flight Results

We utilized data from the newly created IndexP log to train the model and deploy it as a new service in Caique (see Figure 9). The offline model had roughly 4B×5 = 20 B features for the LR model, which was cut to 330M×5 = 1.65 B features for online serving. The daily training job utilizes 20 machines and has a training time of 7 hours per day. The input sample size is 16 B per day and data size is 13 TB per day (we store 90 days of data for the experiment). In the next subsection, we present cross validation and winrate measurement results from the online flight. In the following subsection, we present market level key performance indicators (KPIs) that are important for the RSA product.

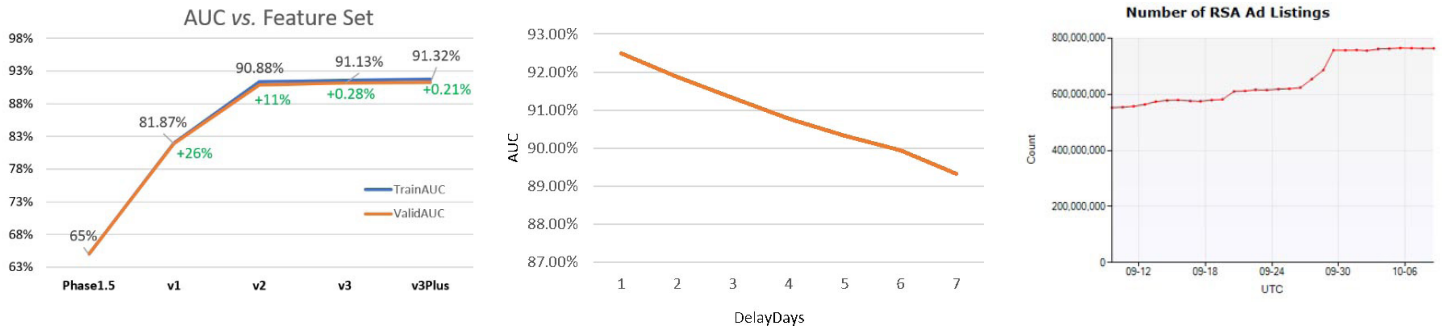


Figure 10: (a) Our model (v1) improves over a model that is completely based on lookups (Phase 1.5). Adding contextual features (v2) lifts this score by an additional 11%. More specific query features (v2, v3 and v3 plus) provide further gain. (b) AUC drops when the same model is used over multiple days because of (c) increase in RSA demand over time.

4.1. AUC and WinRate Measurements

Date/AUC	Phase1.5	Phase2	DiffRatio
01/27/2020	62.33%	84.56%	+35.66%
01/28/2020	62.04%	84.10%	+35.55%
01/29/2020	61.91%	84.95%	+37.22%
01/30/2020	61.92%	84.57%	+36.56%
01/31/2020	62.26%	84.38%	+35.54%
02/01/2020	63.54%	84.85%	+33.54%
02/02/2020	63.80%	84.95%	+33.16%

Table3: Phase 2 provides over 30\% lift in winrate AUC compared to Phase 1.5.

Table 3 shows cross validation results between the control model, which is the log-based asset selection model (Phase 1.5), and the treatment model, which is the proposed approach (Phase 2) on treatment traffic. The treatment model has over 30% overall AUC gain compared to the control model over all 7 days of testing. The AUC sliced by number of ad combinations is shown in Table 4. The impression ratio (ImpRatio) columns shows the percentage of ad samples that appear for specific slices of possible combination count (depending on the number of assets provided by advertisers). The ListingAdRatio gives the percentage of ad-groups and AdvertiserRatio column shows the number of advertisers who fall into each bucket. From the table, we see that the winrate AUC generally reduced as the number of possible combinations increase. This is consistent because as the action space gets larger, both the exploration mechanism and the sequential LR approximation become weaker.

Table 5 shows the sample level winrate that is achieved for the asset combinations produced by different models. The Phase 2 model achieves the highest winrate across all model options that we evaluated. Figure 11 shows the per-advertiser winrate percentage as a CDF computed over ratio of total advertiser count. Advertisers with smaller per-advertiser winrate are at the bottom of the y-axis and advertisers with high per-advertiser winrate are at the top. We observe from the figure that the RSA Phase 2 model has more proportion of advertisers with higher per-advertiser winrate when compared to Phase 1.5 and Phase 1.

Date/Winrate	Phase1	Phase1.5	Phase2
10/21/2019	4.80%	7.59%	7.96% (+4.92%)
10/22/2019	4.90%	7.61%	8.06% (+5.87%)
10/23/2019	4.89%	7.50%	7.92% (+5.53%)
10/24/2019	4.83%	7.39%	7.77% (+5.18%)
10/25/2019	4.77%	7.34%	7.67% (+4.61%)

Table 5: Per sample winrate computed as an average over all RSA samples.

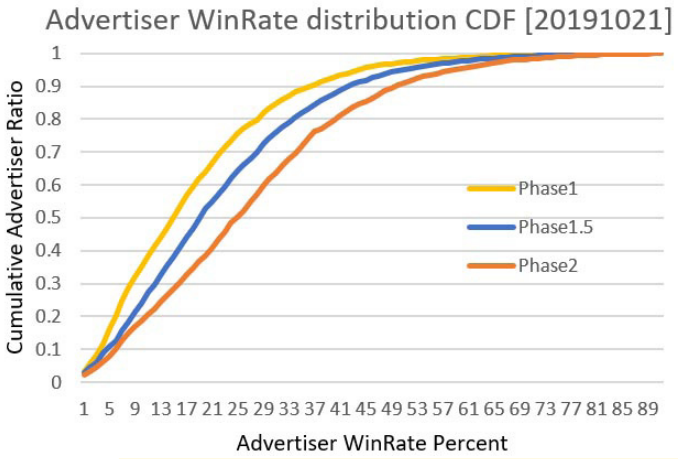


Figure 11: Phase 2 RSA model has the highest value of per-advertiser winrate.

4.2. Market level KPIs

There are several market-place metrics that matter in computational advertising **(Yi et al., 2013)**. Top part of Table 6 shows the flights metrics for RSA Phase 2 model that was shipped into production compared with the Phase 1.5 model. Just looking at the key metrics, we observe that there

Combination	ImpRatio	AdRatio	AdvertiserRatio	Phase1.5 Auc	Phase2 Auc	DiffRatio
ALL	100.00%	100%	100%	61.9%	84.6%	+36.56%
ComCnt < 73	10.85%	4%	17%	62.1%	84.9%	+36.78%
ComCnt $\in [73, 361)$	10.79%	6%	21%	58.6%	83.2%	+41.95%
ComCnt $\in [361; 4032)$	15.14%	7%	30%	57.6%	81.7%	+41.84%
ComCnt $= 4032$	10.56%	15%	6%	79.3%	89.1%	+12.31%
ComCnt $\in [4033; 32760)$	10.12%	7%	18%	56.2%	81.7%	+45.50%
ComCnt $=$ 32760	42.24%	60%	8%	61.1%	84.8%	+38.77%

Table 4: Fewer number of asset combinations per RSA ad show higher AUC lift, which is consistent with the modeling approximations that we make that are directly proportional to the complexity of the action space.

is an increase in MLIY (main-line impression yield), MLCY (main-line click yield) and PxMLIY (main-line impression yield in terms of pixels). Higher value of these metrics demonstrate that more RSA ads are being impressed and clicked more compared to the control model.

The middle part of Table 6 shows that the metrics when we replace the LR model with a DNN and the bottom part shows them when multiple asset combinations are selected and sent to the auction, as opposed to a single combination. In all cases there is a cumulative increase in both click yield and impression yield. In terms of revenue, these translate to increased revenue-per-mille (RPM), as again seen from the tables.

International status. The results shown in Table 6 are for the EN-US market. We have also deployed these models in international markets (EMEA, APAC and others). Table 7 shows the KPIs for these markets and like the EN-US market, these metrics are in line with expectation and are on the positive side.

5. Conclusions

In this white paper, we proposed to solve the problem of asset stitching for a new type of ad product called Responsive Search Ads. We approximate the complex search space through a sequential scoring model, utilize contextual features based on user-issued queries and explore different combination choices via a per-position Thompson Sampling methodology. We demonstrate that the AUC over auction wins and losses for the stitched ads increased by up to 30% compared to a model that is totally count based. Our model enables us to utilize semantic information within the asset text and exploits query-based features to more accurately capture the user intent. By measuring marketplace metrics over the shipped model in production across different geographies, we observe an increase in click yield (CY) and impression yield (IY), which lead to an increase in revenue-per-mille (RPM) for the Bing Ads product.

Next steps. We plan to include user-based features and counting features, and also improve the LR model with a more complex architectures such as a DNN or RNN for predicting the winrate of the stitched asset combinations. Through this model, we have learnt that it is critical to capture query and user intent in the ad-creation process. It is also important to build models that can work well across all geographies.

Flight Name	RPM	CPC	CY	IY	CTR	MLCY	MLIY	PxMLIY	Latency
Phase2 LR VS. Phase1.5	0.22%	0.28%	-0.06%	0.03%	-0.10%	-0.04%	0.03%	0.08%	3.23%
RSA segment	31.26%	22.58%	7.08%	5.76%	1.25%	9.39%	5.52%	32.71%	
Phase2 DNN VS. LR	0.04%	-0.04%	0.07%	0.11%	-0.03%	0.06%	0.11%	-0.04%	0.12%
RSA segment	1.74%	-4.99%	7.08%	7.29%	-0.20%	6.79%	7.01%	1.09%	
PHASE2 LR 2 combination	-0.15%	-0.17%	0.02%	0.03%	0.00%	0.00%	-0.01%	0.09%	0.49%
RSA segment	7.05%	-0.02%	7.07%	7.11%	-0.04%	6.89%	6.52%	8.81%	

Table 6: Phase 2 RSA model shows increase in marketplace KPIs and revenue for Bing Ads. Fields in bold are core metrics we focus on.

Flight Name	RPM	CPC	CY	MLCY	PxMLIY	Latency
[AU,NZ] Phase2 LR VS. Phase1.5	-0.39%	-0.14%	-0.25%	-0.25%	-0.07%	5.82%
RSA segment	11.03%	4.22%	6.54%	7.08%	3.27%	
[SEA] Phase2 LR VS. Phase1.5	1.01%	1.21%	-0.20%	-0.20%	0.07%	2.77%
RSA segment	18.45%	7.48%	10.21%	9.25%	6.55%	
[CN-TW-HK] Phase2 LR VS. Phase1.5	0.97%	0.68%	0.28%	0.34%	0.30%	2.41%
RSA segment	9.05%	1.72%	7.21%	7.19%	7.73%	

Table 7: International models also show increase in KPIs like the EN-US market. Fields in bold are the core metrics we focus on.

References

Abernethy, J. D., Amin, K., Kearns, M. J., and Draief, M. (2013). Large-scale bandit problems and KWIK learning. *In Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013, volume 28 of JMLRWorkshop and Conference Proceedings*, pages 588–596. JMLR.org.

Agrawal, S. and Goyal, N. (2017). Near-optimal regret bounds for thompson sampling. *J. ACM*, 64(5):30:1– 30:24.

Beygelzimer, A., Langford, J., Li, L., Reyzin, L., and Schapire, R. E. (2011). Contextual bandit algorithms with supervised learning guarantees. In Gordon, G. J., Dunson, D. B., and Dudík, M., editors, *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2011, Fort Lauderdale, USA, April 11-13, 2011*, volume 15 of *JMLR Proceedings*, pages 19–26. JMLR.org.

Bietti, A., Agarwal, A., and Langford, J. (2010). A contextual bandit bake-off. *CoRR*, abs/1802.04064.

Chan, D., Ge, R., Gershony, O., Hesterberg, T., and Lambert, D. (2010). Evaluating online ad campaigns in a pipeline: causal models at scale. In Rao, B., Krishnapuram, B., Tomkins, A., and Yang, Q., editors, *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, July 25-28, 2010*, pages 7–16. ACM.

Chapelle, O. and Li, L. (2011). An empirical evaluation of thompson sampling. In Shawe-Taylor, J., Zemel, R. S., Bartlett, P. L., Pereira, F. C. N., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems 2011. Proceedings of a meeting held 12-14 December 2011, Granada, Spain*, pages 2249–2257.

Chu, W., Li, L., Reyzin, L., and Schapire, R. E. (2011). Contextual bandits with linear payoff functions. In Gordon, G. J., Dunson, D. B., and Dudík, M., editors, *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2011, Fort Lauderdale, USA, April 11-13, 2011*, volume 15 of *JMLR Proceedings*, pages 208–214. JMLR.org.

Dudík, M., Langford, J., and Li, L. (2011). Doubly robust policy evaluation and learning. In Getoor, L. and Scheffer, T., editors, *Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011*, pages 1097–1104. Omnipress.

Hillard, D., Schroedl, S., Manavoglu, E., Raghavan, H., and Leggetter, C. (2010). Improving ad relevance in sponsored search. In Davison, B. D., Suel, T., Craswell, N., and Liu, B., editors, *Proceedings of the Third International Conference on Web Search and Web Data Mining, WSDM 2010, New York, NY, USA, February 4-6, 2010*, pages 361–370. ACM.

Li, L., Chu, W., Langford, J., and Schapire, R. E. (2010). A contextual-bandit approach to personalized news article recommendation. In Rappa, M., Jones, P., Freire, J., and Chakrabarti, S., editors, *Proceedings of the 19th International Conference on World Wide Web, WWW 2010, Raleigh, North Carolina, USA, April 26-30, 2010*, pages 661–670. ACM.

McMahan, H. B., Holt, G., Sculley, D., Young, M., Ebner, D., Grady, J., Nie, L., Phillips, T., Davydov, E., Golovin, D., Chikkerur, S., Liu, D., Wattenberg, M., Hrafnkelsson, A. M., Boulos, T., and Kubica, J. (2013). Ad click prediction: a view from the trenches. In Dhillon, I. S., Koren, Y., Ghani, R., Senator, T. E., Bradley, P., Parekh, R., He, J., Grossman, R. L., and Uthurusamy, R., editors, *The 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2013, Chicago, IL, USA, August 11-14, 2013*, pages 1222–1230. ACM.

Shaparenko, B., Çetin, Ö., and Iyer, R. (2009). Data-driven text features for sponsored search click prediction. In Li, Y., Surendran, A. C., and Shen, D., editors, *Proceedings of the 3rd ACM SIGKDD Workshop on Data Mining and Audience Intelligence for Advertising, Paris, France, June 28, 2009*, pages 46–54. ACM.

Yi, J., Chen, Y., Li, J., Sett, S., and Yan, T. W. (2013). Predictive model performance: offline and online evaluations. In Dhillon, I. S., Koren, Y., Ghani, R., Senator, T. E., Bradley, P., Parekh, R., He, J., Grossman, R. L., and Uthurusamy, R., editors, *The 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2013, Chicago, IL, USA, August 11-14, 2013*, pages 1294–1302. ACM.

Zaheer, M., Kottur, S., Ravanbakhsh, S., Póczos, B., Salakhutdinov, R., and Smola, A. J. (2017). Deep sets. In Guyon, I., von Luxburg, U., Bengio, S., Wallach, H. M., Fergus, R., Vishwanathan, S. V. N., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 3391–3401.

Zhang, C., Agarwal, A., III, H. D., Langford, J., and Negahban, S. (2019). Warm-starting contextual bandits: Robustly combining supervised and bandit feedback. In Chaudhuri, K. and Salakhutdinov, R., editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 7335–7344. PMLR.