Prediction and Exploration Models for Responsive Search Ads

Jiamei Shuai, Shuayb Zarar, Denis Charles
{jiashua,shuayb,cdx}@microsoft.com

Microsoft Advertising

1. Introduction

In digital advertising, a text ad (TA) that shows up along with search results, also known as a search-engine ad, is composed of titles, descriptions, typically referred to as assets, and other pieces of information. 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 [1][2]. 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-engine ad. Figure 1 shows an overview of the RSA product. Advertisers provide 3-15 titles and 2-4 descriptions as part of the ad-creative process. These pieces can be combined in approximately 32,000 ways. Each combination of an RSA ad has specific relevance to a query and likelihood of being clicked by a user.

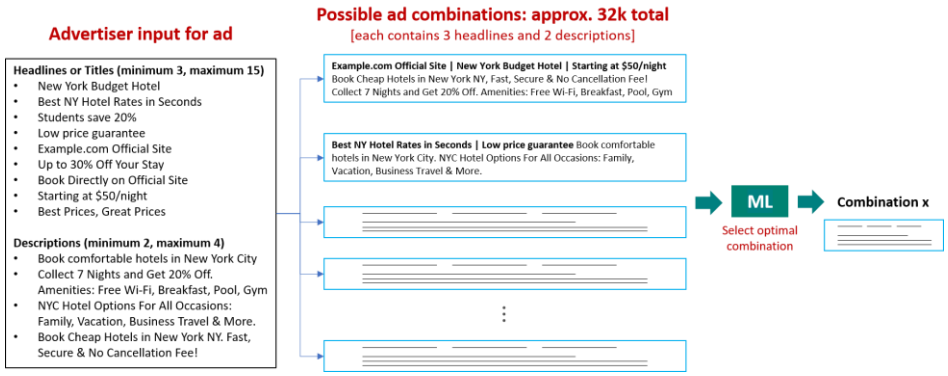

Figure 1. RSA simplifies ad-creation process. It requires advertisers to provide headline (up to 15) and description (up to 4) assets. It utilizes machine-learning to select assets and stitch together an ad-copy depending on the user and query context at runtime

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 [3-6]. We have shipped this model to GA in January 2020, and it currently serves live traffic from bing.com in USA and several international markets. Figure 2 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.

Figure 2. RSA has achieved 6% adoption based on servable customer count and shows uptrends in revenue and other KPIs compared to other ad products such as STA (search TA), EXTA (extended TA) and DSA (dynamic search ads).

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.

2.1. Objective function

To model 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:

$$\text{winrate for asset combination} = \frac{\text{Number of times asset combination is not filtered in auction}}{\text{Number of times asset combination is sent to participate in auction}}$$

winrate is a measure of how often an ad-combination wins an auction.

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 15x14x13x5x4 = 32,760. 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 rest of this section, we present more details about this model.

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. 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: $\mathrm{P}(Y = 1 \,|X; W) = f(X, W) = \sigma(W^T X + b)$

where W is the 5x4B dimensional weight vector that is learnt to minimize the binary cross-entropy loss $\mathrm{J(w)}$ over M training examples as follows: $\mathrm{J(w)} = -\frac{1}{\mathrm{M}} \sum_{i=0}^{M} \mathrm{Y}^{(i)} \log P(Y = 1) + \left(1 - Y^{(i)}\right) \log P(Y = 0)$

At inference time, we perform a sequential per-position asset selection. 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:

$$\mathrm{P}(Y = 1|X_i; W_i) = f(X_i, W_i, b\,) = \sigma\left(W_i^T X_i + b\right), \text{with } i \in [1,5]$$

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 LR 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. We also include an exploration mechanism in the model.

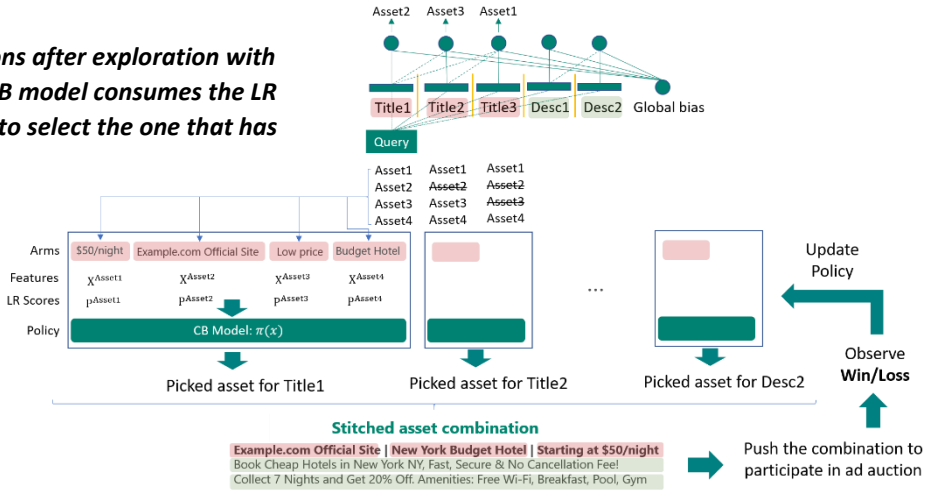

Figure 4. We select asset combinations after exploration with sequential contextual bandits. The CB model consumes the LR scores and features from each asset to select the one that has

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 [7][8]. There are related approaches that solve similar problems through set-level estimation [9]. 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 4. 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, including Ɛ-greedy methods [10], upper confidence bounds (UCB) [11][12], or Thompson Sampling [13][14]. Thompson Sampling is a randomized policy and has been shown to have lower regret [13]. 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 the parameters of the Beta distribution that allow us to control the level exploration and exploitation that can be achieved during the decision-making process. *N* is the number of observations that have been made for the reward function at any point of time in the sequential decision-making process: auction wins plus losses. Smaller values of α, β and N (computed using gradient sum of the LR model) lead to more exploration and larger values lead to exploitation of the current logistic regression model.

3. Offline Experimental Evaluation

An overview of the experimental framework is shown in Figure 5. We utilize historical logs to train the LR model. We create a new log called IndexP log, which records asset combinations that do not win in the auction; earlier, the existing IndexP log stored only those asset combinations that won the auction which ran in Listing Server. 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 components besides the ads auction.

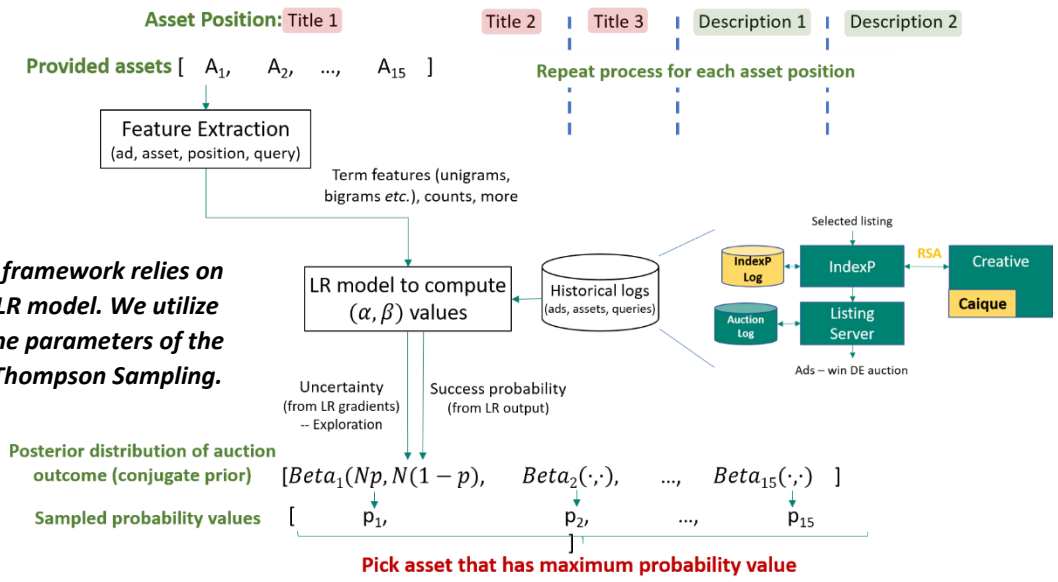

Figure 5. Our experimental framework relies on historical logs to train the LR model. We utilize the LR model to estimate the parameters of the Beta Distributions used in Thompson Sampling.

Offline winrate simulation. When evaluated in isolation, the proposed model has selection bias because the ad-level features are correlated with both sample selection and the outcome of the model. A common strategy to remove this bias is propensity score weighting [4][16]. Basically, we re-weight data 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 for each ad $\mathrm{Ad_i}$,that is represented by up to 15 title and 4 description assets, in our system in the form of $\mathrm{D_i} = \{(\mathrm{x_1}, \mathrm{a_1}, \mathrm{p_1}, \mathrm{r_1}), \cdots, (\mathrm{x_n}, \mathrm{a_n}, \mathrm{p_n}, \mathrm{r_n})\}$, where $x_i = (x^{Combination_1}, \ldots, x^{Combination_n})$ are the features corresponding to different asset combinations that for the same ad $\mathrm{Ad_i}$; $\mathrm{a_i}$ is an asset combination from various asset options in this ad for which an action is logged in the system (that is win or loss), $\mathrm{p_i}$ is the probability that the ad is selected by the policy w to create an asset combination and $\mathrm{r_j}$ is the reward obtained when this asset combination was produced, which is auction win or loss in our case. For ad $\mathrm{Ad_i}$ that is processed by policy w, we can compute the IPS score as follows: $IPS_{Ad_i}(\pi_w) = \sum_{j=1}^{n} [\frac{r_j}{p_j} P(\pi_w(x_j) = a_j)]$

where $\pi_w(x_j)$ is the maximum value of the predicted win rate for all asset combinations from that ad, and $\mathrm{P}(\pi_w(x_j) = a_j)$ is the probability that asset a_j had that maximum value. r_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) 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 9, 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 1, 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.

Table 1. LR+CB model achieves the highest per advertiser winrate across 100k ad combinations that we evaluated.

Model	ListingAdCnt	PerAd WinRate	PerAd WinRate Delta	PerAd WinRate Delta StdDev
Phase 1	103,071	8.30%	0.00%	NA
Phase 1.5	103,071	8.48%	2.13%	0.072%
Phase 2	103,071	8.60%	3.59%	0.074%

3.1. Prediction model results

In this section, we discuss impact of different hyperparameters on the modeling objective. Figure 9 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 sequential LR+CB 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, v3 and v3Plus) brings an additional AUC improvement of 11%. We observed that tuning *l1* parameters in FTRL [17] leads to a smaller model size with little impact on the AUC. For instance, adding l1=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 entire data corpus takes about 7 hrs. Training a second epoch takes the same amount of time but provides a 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 9 (b) shows the AUC drop when we use the same model over multiple days. And Figure 9 (c) shows the change in demand over multiple days.

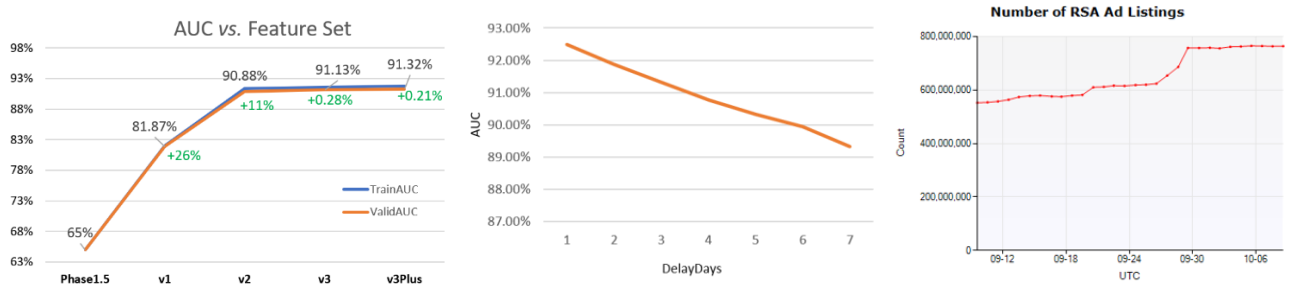

Figure 9. (a) Our model (v1) improves over a model that is completely based on lookups (Phase 1.5). Adding contextual features (v2) lifts this score by an additional 11%. More specific query features (v23 and v3 plus) provide further gain. (b) AUC drops when the same model is used over multiple days because of (c) increase in RSA demand over time.

4. 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 8). The offline model had roughly 4B x 5 = 20 B features for the LR model, which was cut to 330M x 5 = 1.65 B features for online serving. The daily training job utilizes 20 machines and has a training time of 7 hrs. per day. The input sample size is 16 B per day and data size is 13 B 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.

4.1. Win Rate Measurements

Table 2. Phase 2 provides over 30% lift in winrate AUC compared to Phase 1.5

	F41315 (Phase2 Treatment)		
	AUC		AUC_DiffDelta
Date	Phase1.5	Phase2	Phase2
20200127	62.33%	84.56%	**35.66%**
20200128	62.04%	84.10%	**35.55%**
20200129	61.91%	84.95%	**37.22%**
20200130	61.92%	84.57%	**36.56%**
20200131	62.26%	84.38%	**35.54%**
20200201	63.54%	84.85%	**33.54%**
20200202	63.80%	84.95%	**33.16%**

Table 2 shows cross validation results between the control model (Phase 1.5), and the treatment models (Phase 2). 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 3. The impression ration (impRatio) columns shows the percentage of ad impressions that appear for different number of possible combinations (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 4 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 10 shows the per-advertiser winrate as a CDF computed over advertiser volume. Small advertisers are at the bottom of the y-axis and larger advertisers (advertisers with high RSA ad volume) are at the top. We observe from the figure that the RSA Phase 2 model does well over all types of advertisers when compared to Phase 1.5 and Phase 1.

Table 3. 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.

20200130 Row Labels	ImpRatio	ListingAdRatio	AdvertiserRatio	Phase1.5 AUC	Phase2 AUC	AUC_DiffRatio
FlightId=41315	100.00%	100%	100%	61.9%	84.6%	36.56%
FlightId=41315,ComCnt_<73	10.85%	4%	17%	62.1%	84.9%	36.78%
FlightId=41315,ComCnt_>=_73_And <361	10.79%	6%	21%	58.6%	83.2%	41.95%
FlightId=41315,ComCnt_>=_361_And <4032	15.14%	7%	30%	57.6%	81.7%	41.84%
FlightId=41315,ComCnt_>=_4032_And <4033	**10.56%**	**15%**	**6%**	**79.3%**	**89.1%**	**12.31%**
FlightId=41315,ComCnt_>=_4033_And <32760	10.12%	7%	18%	56.2%	81.7%	45.50%
FlightId=41315,ComCnt>=32760	**42.24%**	**60%**	**8%**	**61.1%**	**84.8%**	**38.77%**

Table 4. Per sample winrate computed as an average over all RSA samples

Date/WinRate	Phase1 (Random)	Phase1.5	Phase2	Phase2_DiffRatio
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%

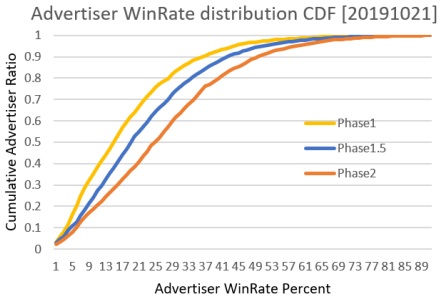

Figure 10. Per-advertiser winrate.

4.2. Market level KPIs

There are several market-place metrics that matter in computational advertising [15]. Top part of Table 5 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 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 5 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 6 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.

Table 5. Phase 2 RSA model shows an increase marketplace KPIs as well as the revenue for Bing Ads.

	Flight Name	Pub	Days	Trmt	Ctrl	RPM	CPC	CY	IY	CTR	Cov	MLCY	MLIY	PxMLIY	DE_Latency Avg
8	RSA phase2 DNN	All	13.6	86953	41315	0.04 %	-0.04 %	0.07 %	0.11 %	-0.03 %	0.10 %	0.06 %	0.11 %	-0.04 %	0.12 %
9	RSA phase2 DNN [RSA]	All	13.6	86953	41315	1.74 %	-4.99 %	7.08 %	7.29 %	-0.20 %	6.77 %	6.79 %	7.01 %	1.09 %	

	Flight Name	Pub	Days	Trmt	Ctrl	RPM	CPC	CY	IY	CTR	Cov	MLCY	MLIY	PxMLIY	DE_Latency Avg
14	RSA Phase2 (2 comb) -- Higher exploration (2.4%)	All	8.5	89400	41315	-0.15 %	-0.17 %	0.02 %	0.03 %	0.00 %	0.04 %	0.00 %	-0.01 %	0.09 %	0.49 %
15	RSA Phase2 (2 comb) -- Higher exploration [RSA]	All	8.5	89400	41315	7.05 %	-0.02 %	7.07 %	7.11 %	-0.04 %	6.49 %	6.89 %	6.52 %	8.81 %	

	Flight Name	Pub	Days	Trmt	Ctrl	RPM	CPC	CY	IY	CTR	Cov	MLCY	MLIY	PxMLIY	DE_Latency Avg
RSA Phase 2															
10	RSA phase2 IndexPLog [3%] vs. phase1.5	All	9.4	83873	41315	0.22 %	0.28 %	-0.06 %	0.03 %	-0.10 %	-0.01 %	-0.04 %	0.03 %	0.08 %	3.23 %
11	RSA phase2 IndexPLog [3%] vs. phase1.5 [RSA]	All	9.4	83873	41315	31.26 %	22.58 %	7.08 %	5.76 %	1.25 %	5.06 %	9.39 %	5.52 %	32.71 %	

Table 6. International models also show increase in KPIs similar to the EN-US market.

	Flight Name	Pub	Days	Trmt	Ctrl	RPM	CPC	MLCY	CY	PxMLIY	DE_Latency Avg
3	[AU, NZ] RSA Phase 2.0 (US model)	All	9.4	87053	53323	-0.39 %	-0.14 %	-0.25 %	-0.25 %	-0.07 %	5.82 %
4	[AU, NZ] RSA Phase 2.0 (US model) [RSA]	All	9.4	87053	53323	11.03 %	4.22 %	7.08 %	6.54 %	3.27 %	
5	[SEA] RSA Phase 2.0 (Us Model)	All	9.4	87457	53313	1.01 %	1.21 %	-0.20 %	-0.20 %	0.07 %	2.77 %
6	[SEA] RSA Phase 2.0 (Us Model) [RSA]	All	9.4	87457	53313	18.45 %	7.48 %	9.25 %	10.21 %	6.55 %	
7	[CN-TW-HK] RSA Phase2.0 (US Model)	All	9.4	87458	53320	0.97 %	0.68 %	0.34 %	0.28 %	0.30 %	2.41 %
8	[CN-TW-HK] RSA Phase2.0 (US Model) [RSA]	All	9.4	87458	53320	9.05 %	1.72 %	7.19 %	7.21 %	7.73 %	

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 not data driven. 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 historical click data and 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.

6. References

[1]. D. Hillard, S. Schroedl, E. Manavoglu, H. Raghavan, Chris Leggetter: Improving ad relevance in sponsored search. WSDM 2010: 361-370

[2]. B. Shaparenko, O. Çetin, R. Iyer: Data-driven text features for sponsored search click prediction. KDD Workshop on Data Mining and Audience Intelligence for Advertising 2009: 46-54

[3]. A. Beygelzimer, J. Langford, L. Li, L. Reyzin, and R. E. Schapire. Contextual bandit algorithms with supervised learning guarantees. In AISTATS, 2011.

[4]. M. Dudík, J. Langford and L. Li. Doubly robust policy evaluation and learning. In ICML,2011b.

[5]. A. Bietti, A. Agarwal and J. Langford, A contextual bandit bake-off. arXiv preprint arXiv:1802.04064, 2018.

[6]. C. Zhang, Alekh Agarwal, Hal Daumé III, John Langford, Sahand Negahban: Warm-starting Contextual Bandits: Robustly Combining Supervised and Bandit Feedback. ICML 2019: 7335-7344

[7]. https://www.microsoft.com/en-us/research/blog/contextual-bandit-breakthrough-enables-deeper-personalization/, Miro Dudík

[8]. J. D. Abernethy, K. Amin, M. J. Kearns, M. Draief: Large-Scale Bandit Problems and KWIK Learning. ICML (1) 2013: 588-596

[9]. M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Póczos, R. Salakhutdinov, A. J. Smola: Deep Sets. NIPS 2017: 3391-3401

[10]. https://lilianweng.github.io/lil-log/2018/01/23/the-multi-armed-bandit-problem-and-its-solutions.html, Lilian Weng

[11]. L. Li, W. Chu, J. Langford, R. E. Schapire: A contextual-bandit approach to personalized news article recommendation. WWW 2010: 661-670

[12]. W. Chu, L. Li, L. Reyzin and R. E. Schapire, Contextual bandits with linear payoff functions. In Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2011, Fort Lauderdale, USA, April 11-13, 2011, pp. 208–214, 2011. URL http://www.jmlr.org/proceedings/papers/v15/chu11a/chu11a.pdf.

[13]. S. Agrawal, N. Goyal: Near-Optimal Regret Bounds for Thompson Sampling. J. ACM 64(5): 30:1-30:24 (2017)

[14]. O. Chapelle, L. Li: An Empirical Evaluation of Thompson Sampling. NIPS 2011: 2249-2257

[15]. J. Yi, Y. Chen, J. Li, S. Sett, T. W. Yan: Predictive model performance: offline and online evaluations. KDD 2013: 1294-1302

[16]. D. Chan, R. Ge, O. Gershony, T. Hesterberg, D. Lambert, Evaluating Online Ad Campaigns in a Pipeline: Causal Models At Scale, Proceedings of ACM SIGKDD 2010, pp. 7-15

[17]. H. B. McMahan, G. Holt, D. Sculley, M. Young, D. Ebner, J. Grady, L. Nie, T. Phillips, E. Davydov, D. Golovin, S. Chikkerur, D. Liu, M. Wattenberg, A. M. Hrafnkelsson, T. Boulos, J. Kubica: Ad click prediction: a view from the trenches. KDD 2013: 1222-1230