Estimating Ad Group Performance in Sponsored Search

Dawei Yin[†] Bin Cao[S] Jian-Tao Sun[S] Brian D. Davison[†]
[†]Dept. of Computer Science and Engineering, Lehigh University, Bethlehem, PA, USA
[S]Microsoft Research Asia, Beijing, China
[†]{day207, davison}@cse.lehigh.edu [S]{bincao, jtsun}@microsoft.com

ABSTRACT

In modern commercial search engines, the pay-per-click (PPC) advertising model is widely used in sponsored search. The search engines try to deliver ads which can produce greater click yields (the total number of clicks for the list of ads per impression). Therefore, predicting user clicks plays a critical role in sponsored search. The current ad-delivery strategy is a two-step approach which first predicts individual ad CTR for the given query and then selects the ads with higher predicted CTR. However, this strategy is naturally suboptimal and correlation between ads is often ignored under this strategy. The learning problem is focused on predicting individual performance rather than group performance which is the more important measurement.

In this paper, we study click yield measurement in sponsored search and focus on the problem—predicting group performance (click yields) in sponsored search. To tackle all challenges in this problem—depth effects, interactive influence, cold start and sparseness of ad textual information—we first investigate several effects and propose a novel framework that could directly predict group performance for lists of ads. Our extensive experiments on a large-scale real-world dataset from a commercial search engine show that we achieve significant improvement by solving the sponsored search problem from the new perspective. Our methods noticeably outperform existing state-of-the-art approaches.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval; H.2.8 [**Database Management**]: Database applications—*Data Mining*

Keywords

Sponsored search, CTR, Click yield, Ad clicks

1. INTRODUCTION

In modern commercial search engines, the pay-per-click (PPC) advertising model is widely used in sponsored search. Search engines try to deliver ads which can produce greater click yields (the total number of clicks for the list of ads per impression) for more revenue. Therefore, predicting user clicks plays a critical role in online services, which helps the system to serve items with high CTRs.

Current ad-delivery strategies are two-step approaches, and work as follows. The system first predicts individual ad click-through rates (CTRs) for the given query. Based on the estimated CTRs, the system selects the ads with high predicted CTR, subject to some conditions such as user utility and advertiser ROI [4]. The underlying assumption is that the rewards of ads are independent and clicks on the ads in a list is additive. However, this assumption does not hold in reality. For example, two similar ads in the same list could reduce the CTRs of each [39]. Moreover, this strategy is naturally suboptimal. The learning problem is focused on predicting individual performance rather than group performance which is selecting the best list of ads to maximize overall payoff. Under this scheme, the overall performance (e.g., click yield—the total number of clicks for the list of ads per impression) becomes another important measurement for ad selection in sponsored search. In this work, we consider click yield as the measure of group performance. The motivation behind the proposal to change the objective is based on nontrivial observations. That is, two systems that have similar precision on predicting individual performance could behave very differently on group performance. For example, we can have two models having the same CTR prediction accuracy. However, the ads selected by the two models have huge differences on the clicks obtained. Moreover, since the overall click yields are based on the delivered ad list and the CTRs are not independent, the results could be suboptimal even when the individual ad CTR prediction is perfect. In fact, it is not necessary to know accurately the CTRs of each individual ad, as long as we can select the best lists. Sometimes estimating CTRs of all ads may be an even more challenging problem due to the problem of sparsity.

Here, we challenge the traditional strategy for predicting group performance based on CTR prediction. We propose a novel framework that could directly predict the click yield for a ranked list of ads which has rarely been studied but is an important measurement for ad delivery. The problem is essentially a ranking problem since we would like to judge which lists of ads are better. However, unlike the problem of learning to rank that ranks single documents, here we rank sets of ads. Additionally, we leverage a special constraint on the problem. That is, the number of ads shown on a page is usually limited to a small number, (e.g., four mainline ads in a typical commercial search engine). Still, it is not realistic to explore all possible combination of ads. Therefore, we simplify the problem to rank historically presented ad lists only (that is, the ad lists which have been shown before).

WSDM'14, February 24–28, 2014, New York, New York, USA.

http://dx.doi.org/10.1145/2556195.2556257 .

However, solving the click yields prediction/ranking problem is still nontrivial for several reasons. At first, unlike organic search, sponsored search has its own bias and characteristics. The probability of a user examining sponsored search results is related to two major factors: the depth of the ads (the number of ads displayed) and the commercial intent of a query. For the depth of the ads, usually the more ads are shown at the same time, the larger the ad area is and the more attention from users is attracted. Moreover, with too many ads displayed together, the overall effects on click yields may not be always positive. On the other hand, for some queries with high commercial intent, ads usually are interesting to users, while for other queries, showing ads may annoy users. Secondly, interactive influences between query and ad lists and across ads/positions are also important factors to the click yields problem. Some preliminary work [32, 39] on CTR prediction has shown the existence of these interaction effects. However, these studies have focused on independent ads, and the effects of interactive influences and position correlation on total click yield are still not clear. Thirdly, cold start problems (new ads) also occur frequently in sponsored search [30, 36]. To model and utilize the content of the ad is critical when tackling this problem, but few investigations of ad content (short and sparse texts) have been conducted.

To estimate the ad group performance in sponsored search and tackle these challenging problems, we combine heterogenous types of information into the learning model. We first investigate several effects including latent biases, contextual features, interactive influence and correlation over positions. We find that a unified model can achieve the best performance. To best leverage the text features and solve the sparseness issue in text information and cold start on ads, we embed the topic coding model into our framework to learn the topic information of short ad text. We also discuss the effects of various loss functions. We find that ranking loss is preferred for this problem. Finally, we conduct extensive experiments on a large-scale real-world dataset from a commercial search engine. Our methods achieve significant improvement by solving the sponsored search problem from this new perspective.

We summarize our contributions as follow,

- We introduce a novel perspective for sponsored search—click yield—which measures the group performance of ads displayed together.
- To solve the click yield prediction problem, we propose a Click Yield Prediction framework which incorporates multiple factors, such latent bias, textual features, interactive influences, and position correlation.
- Our experiments, based on a large-scale real-world dataset, show our methods make noticeable improvements and outperform state-of-the-art methods based on the traditional strategy.

The paper is organized as follows: Section 2 discusses related work and we then present preliminary data analysis in Section 3. Section 4 presents the proposed model, followed by describing an efficient and scalable approach developed for estimating the model parameters. Section 6 presents the empirical evaluations of the proposed approach on a large real-world data set. Section 7 concludes the paper.

2. RELATED WORK

Related work is primarily in three topics: the users' click model, sponsored search and learning to rank techniques.

The bias in user click behaviors during search was first studied through eye-tracking [20]. There are two major assumptions in click models: the Examination Hypothesis [13, 24, 23, 33] and the Cascade Model. The examination hypothesis assumes that if a displayed url is clicked, it must be both examined and relevant [30]. Following the examination hypothesis, there are three basic models: the click over expected clicks (COEC) model [42], the examination model [6] and the logistic model [10]. They are compared in [6] and experimentally demonstrated to be outperformed by the cascade model, proposed by Craswell et al. [10, 16], which assumes that the user views search results from top to bottom and decides whether to click each url. Once a click is issued, documents below the clicked result are not examined regardless of the position. The click chain model [15] and dynamic Bayesian network [6] which is inferred through [27] made a further generalization by allowing the user to abandon examination of more results. The general click model [46] treats all relevance and examination effects in the model as random variables. In addition to the above methods, there have been several click models following the above two assumptions but apply them in different contexts, such as federated search [7], the task centric click model [43], and the position normalized click model [8]. However, the above methods are mainly designed for understanding users' click behaviors on organic search. They fail to explore the specific properties in sponsored search as discussed above. Moreover, they cannot solve the click yield problem.

For modeling sponsored search and predicting ad click through rate, Thore et al. [14] describe a Bayesian model which is based on a probit regression model. Richardson et al. [30] present a method based on logistic regression, where they extract features in four aspects: term CTR, ad quality, order specificity and external sources. In [42], Zhang and Jones introduce the Clicks Over Expected Clicks (COEC) model. Cheng et al. [9] develop user-specific and demographic-based features that reflect the click behaviors of individuals and groups to improve accuracy over the baseline model. Menon et al. [26] view the CTR prediction problems as a problem of matrix completion. Shen et al. [32] propose a personalized model. Recently, Xu et al. investigate the relational click behaviors [40]. They fail to model the more general cases (e.g., 3 or 4 ads) and handle new ads and queries. Xiong et al. [39] shows the mutually exclusive influence between similar ads. These methods are focused on predicting the click-through rate of ads. Although they have been used in selecting ads to display for specific query sessions, they are essentially two-step approaches and do not directly estimate ad group performance.

Another area of related work is learning to rank techniques. In Information Retrieval (IR), a generic task is to construct a ranked list of documents relevant to a query issued by a user. Although ranking is a fundamental problem in IR and has been studied for decades, it still remains challenging. Recent methods are summarized in [25]. In the standard Learning to Rank setting, a typical training set consists of queries with their associated documents represented by feature vectors as well as corresponding relevance judgements. A machine learning algorithm is employed to learn the ranking model, which can predict the ground truth label in the training set as accurately as possible in terms of a loss function. In the test phase, when a new query comes in, the learned model is applied to sort the documents according to their relevance to the query, and return the corresponding ranked list to the user as the response to the query. Depending on different hypotheses, input spaces, output spaces and loss functions, approaches to LtoR can be loosely grouped into three categories [25]: point-wise, pairwise, and list-wise [28].

3. PRELIMINARIES

In this section, we conduct some exploratory experiments to verify our intuitions and formalize the problem definition. We collect data from a commercial search engine in the U.S. English market in April 2012. In total, there are 127,230,609 query sessions, where a query session consists of an input query, a list of sponsored search results and a list of clicked ads with time stamps. Since click data are noisy, we filter out low frequency queries and ads to remove some noise and use the more confident click data for analysis.

3.1 Group Performance

Compared to individual performance, (e.g., individual ad CTR), ad group performance is much closer to the real system performance. For instance, in sponsored search, given a query, we can decide which group of ads to display, according to the estimated group performance.

3.1.1 Click Yield

We define the concept of *click yield* (CY) as follows. Given a query q and an ad list $\mathbf{a} = \{a_1, a_2, \cdots, a_{|\mathbf{a}|}\}$, the click yield is the ratio of the total number of clicks on the ads in $\mathbf{a}$ over the total number of impressions:[1]

$$y_{q,\mathbf{a}} = \frac{\sum_{i=1}^{|\mathbf{a}|} \text{Click}(q, a_i)}{\text{Impr}(q, \mathbf{a})}$$

where $\text{Click}(q, a_i)$ denotes the total number of clicks on the ad a_i, with issued query q and $\text{Impr}(q, \mathbf{a})$ denotes the number of times showing the ad group $\mathbf{a}$, with issued query q. The concept of click yield differs from the click-through rate (CTR) in that it measures the overall performance of ad lists rather than single ads. We see that unlike the click-through rate of an individual ad, $y_{q,\mathbf{a}}$ might be larger than one.

3.1.2 Problem Definition

With the definition of group performance, we formalized the problem as follows: the records are presented by $\mathcal{D} = (q_m, \mathbf{a}_m, y_m)_{m=1}^{M}$. For a single click log entry $(q, \mathbf{a}, y) \in \mathcal{D}$, q is the query in the record, $\mathbf{a}$ is the list of ads $\mathbf{a} = \{a_1, a_2, \cdots, a_{|\mathbf{a}|}\}$ and $|\mathbf{a}|$ refers to the number of ads (i.e., depth). y means the corresponding click yield. In this work, we will try to solve the following problem: given a query q and a collection of available ad lists $\mathcal{D}_q$, we predict the click yields of the ad lists or rank the ad lists and find the best ad list out of $\mathcal{D}_q$ that has the highest click yield.

3.2 Click Yields Analysis

A commercial search engine typically adopts a two-step approach. That is, the search engine first predicts the click-through rate of ads given the query. Then, it selects the ads with higher predicted CTR to display. However, this approach is suboptimal even when the CTR prediction is perfect since it assumes the CTR of individual ads are independent of each other and clicks are additive. In the following analysis we will see this assumption does not hold in reality.

Fig. 1 shows the average click yields of specific depths (number of ads displayed together) over all queries. We can see that as the depth grows, the increase in click yield slows. For instance, the click yield of depth 1 is 0.1798, but the the click yields of depth 2 is only 27% more than depth 1. This shows that with more ads displayed the clicks on individual ads may be diluted. As we can see, the average click yield becomes larger with increasing depth. However, showing more ads does not *always* get the best click yields. We find many examples like the one in Table 1: for query "worldofwatches", if we only display a single ad "http://WorldofWatches.com", the click yield is 0.2096, while if we display this ad with two other watch-related ads, the click yield of those ads is only 0.1428, which is even lower than the single ad. That is because with query "worldofwatches", although the users are interested in watches, the users' intent is quite clear and exactly want the website "http://WorldofWatches.com". Showing more ads may annoy users, and they are more likely to skip the ads area. In the case that some specific ad exactly matches the user's intent, it appears that less is more.

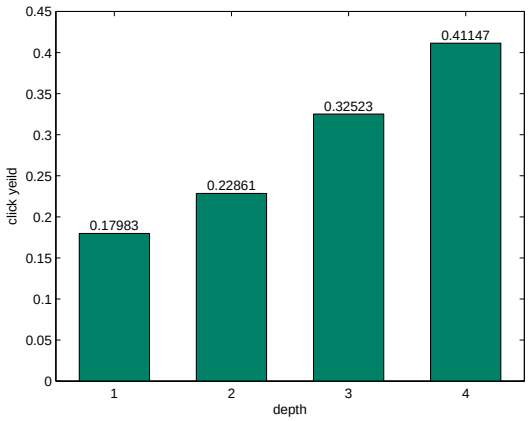


Figure 1: Average Click Yields

We further investigate the effects of the query's commercial intent on click yield and CTR. We treat the average CTR of the query as the measurement for commercial intent of the query (higher average CTR means higher commercial intent), and investigate the relationship between the depth and query commercial intent. In our experiments, we let the depth change from 1 to 3 and then test whether the commercial intent affects the changes of CTR or CY. We categorize the queries by two metrics, CTR and CY. Then, we obtain queries of four types: CTR increasing query, CTR decreasing query, CY increasing query, and CY decreasing query. For the majority of queries, with a larger depth, the average CTR of queries are diluted while the click yields increase, but for some special queries, the average CTR of a large depth is greater than a small depth, or the click yields of a small depth can be larger than that of a large depth. In Figures 2(a) and 2(b), we see that the majority of queries have higher commercial intent (average CTR). In our experiments, there are around 1/10 queries whose average CYs decrease when changing depth from 1 to 3. We see that the commercial intents of these queries are relatively low. Note that even if the average CY of a query increases when changing depth 1 to 3, the gain still may not be linear; that is, one cannot simply use additive CTRs to estimate CY. On the other hand, unfortunately, another type of the unusual cases—CTR increasing (CY increasing obviously) queries—also shares the same property (low commercial intent). That means, we cannot simply use average CTR of ads to decide the number of ads displayed together. A more sophisticated model is necessary to solve this problem. These factors are

[1]The definition of click yield is slightly different from the one used in some other papers [18], where it is defined by the total number of clicks divided by the total number of search result page views. In Hillard et al. [18], the click yield is only a global metric while in this paper it is a metric for ad lists.

Table 1: An example of query "worldofwatches"

Ad list 1 (Click Yields 0.2096)
world of watches (*http://WorldofWatches.com*) Up To 80% Off On The World's Finest Watches. Free Shipping, Shop Now!
Ad list 2 (Click Yields 0.1428)
world of watches (*http://WorldofWatches.com*) Up To 80% Off On The World's Finest Watches. Free Shipping, Shop Now! **Ashford - Luxury for Less** (*http://Ashford.com/Watches_Sale*) Shop Top Brand Watches. Free Shipping. 100%Authentic w/Warranty. **Invicta up to 90% off** (*http://www.nextdaywatches.com*) Authorized Invicta Dealer. Large Selection. Free, Next Day Delivery.

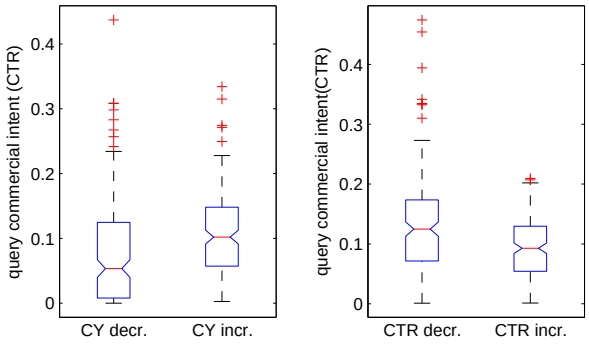


Figure 2: Depth changed from 1 to 3

rarely studied and in the following sections, we will directly study overall performance in ad delivery procedure.

4. PREDICTING CLICK YIELDS

Estimating ad group performance (click yield) is a more complicated problem than individual ad performance, due to extra factors: the number of ads, interactive influences between query and ads, interactive influences across ads. In this section, we will propose a unified model for predicting the click yield of $\mathbf{a}$, given query q. $y_{q,\mathbf{a}}$ represents the click yield. $f_{q,\mathbf{a}}$ the relevance score which can be used for estimating click yield or ranking the ad lists.

4.1 Estimation Model

Latent Bias (LB): One straightforward intuition is that whether the ad is clicked or not depends on the average click rate of the ads: $f(q,\mathbf{a}) = \sum_{i=1}^{k} \mu_{a_i}$ where μ_{a_i} is the CTR of the target ad across all other factors. Certainly, this estimation is too coarse and inaccurate, because as we mentioned before, the majority of ads are not clicked and multiple contextual factors of the ads are not incorporated. We can extend this basic estimation by incorporating wider biases from multiple factors: 1) Positional Query bias, because for each query, the average CTRs on specific positions are different. Given a query q, for each position i, one has a latent bias $\mu_q^{(i)}$. The latent bias of query q can be expressed in a vector form $\boldsymbol{\mu}_q = \left[\mu_q^{(1)}, \mu_q^{(2)}, \cdots, \mu_q^{(d)}\right]^T$ where d denotes the max depth of the ads area (the maximum number of ads shown in ads area). 2) For ad list $\mathbf{a}$, the average CTR of a specific ad depends on the both the position of the ad and the ad itself. Similarly, one has a positional bias for an ad list, $\boldsymbol{\mu}_{\mathbf{a}} = \left[\mu_{a_1}^{(1)}, \mu_{a_2}^{(2)}, \cdots, \mu_{a_d}^{(d)}\right]^T$, where $\mu_a^{(i)}$ denotes the bias of ad a on position i. To compose the final bias of a pair $(q,\mathbf{a})$, an indicator $\mathcal{I}_{\mathbf{a}}$ is also necessary. $\mathcal{I}_{\mathbf{a}}$ points out the positions which are filled by ad list $\mathbf{a}$. We define $\mathcal{I}_{\mathbf{a}} = \left[I_{\mathbf{a}}^{(1)}, I_{\mathbf{a}}^{(2)}, \cdots, I_{\mathbf{a}}^{(d)}\right]^T$ the position indicator vector of $\mathbf{a}$, where $I_{\mathbf{a}}^{(i)} = 1$ if some ad appears on position i otherwise $I_{\mathbf{a}}^{(i)} = 0$. For instance, if $d = 4$ and $|\mathbf{a}| = 2$ which means the first two positions are filled by two ads, then $\mathcal{I}_{\mathbf{a}} = [1,1,0,0]^T$. Then the latent bias is

$$f_{q,\mathbf{a}}^{LB} = \mu_0 + \mathcal{I}_{\mathbf{a}}^T \boldsymbol{\mu}_q + \mathcal{I}_{\mathbf{a}}^T \boldsymbol{\mu}_{\mathbf{a}}$$

where μ_0 is the global bias. Note that these biases are the latent variables to be learned from the data sets. However, this model is appealing since no extra information is needed for learning, besides requiring indicators.

Feature Model (FM): A more powerful model than the bias model is the feature model, which predicts the relevance and harvests the information of features by a linear function of features. For a search session, we could extract the features of the ads and their contexts. Given a record $(q,\mathbf{a})$ and click yields $y_{q,\mathbf{a}}$, the feature vector can be extracted from two aspects:

- **Query Features:** Let $\mathbf{x}_q$ be the feature vector of query q, which can be extracted from the terms of the query or latent topic distribution of the query.
- **Ad Features:** Let $\mathbf{x}_{a_i}$ be the feature vector of ad a_i, which can be extracted from the terms of the shown title and body of the ad, the historical click over expected clicks (COEC) [42] or latent topic distribution of the ad content.

Like the latent bias model, with position and depth information, we define $\mathbf{x}_{\mathbf{a}} = \left[I_{\mathbf{a}}^{(1)}\mathbf{x}_{a_1}^T, I_{\mathbf{a}}^{(2)}\mathbf{x}_{a_2}^T, \cdots, I_{\mathbf{a}}^{(d)}\mathbf{x}_{a_d}^T\right]^T$. A simple linear combination of query features and ad features can be defined as:

$$f_{q,\mathbf{a}}^{FM} = \mathbf{b}_q^T \mathbf{x}_q + \mathbf{b}_{\mathbf{a}}^T \mathbf{x}_{\mathbf{a}}$$

$\mathbf{b}_q$ and $\mathbf{b}_{\mathbf{a}}$ are coefficients to be learned from the training set. The model is essentially equivalent to the one where ad feature and context features are combined into a single feature vector $\mathbf{x} = [\mathbf{x}_q^T, \mathbf{x}_{\mathbf{a}}^T]^T$ and a coeffecient $\mathbf{b} = [\mathbf{b}_q^T, \mathbf{b}_{\mathbf{a}}^T]^T$. Here, we further place a zero mean Gaussian prior or Laplace prior on the values of coefficient $\mathbf{b}$, corresponding to the L_2 and L_1 regularization, respectively. With the Gaussian prior, we have: $\mathbf{b} \sim \mathcal{N}(0, \lambda_{\mathbf{b}}^{-1}\mathbf{I})$

Interactive Influence (IM): Although linear models are efficient, they are usually over-simplified and cannot capture interactions between queries and ads. The interaction between the query and ads are studied in Xiong et al. [39]. However, their methods did not consider to model the interaction across ads. Moreover, their models still are designed for CTR prediction rather than click yield prediction. Similar to Rendle [29], we model interactions of click yields of a query-ad list pair $(q,\mathbf{a})$ through an additive function of ad positions. Let $\mathbf{G}^{(i)} \in \mathbb{R}^{N_a \times N_q}$ be the interaction matrix on position i. The interaction can be modeled as follows.

$$f_{q,\mathbf{a}}^{IM} = \sum_{i=1}^{d} I_{\mathbf{a}}^{(i)} G_{a_i q}^{(i)} \tag{1}$$

The interaction matrix $\mathbf{G}^{(i)}$ is query-position-ad dependent, which is unknown and their entries could be either positive or negative. We treat them as the latent variables to be learned from the data set. However, the model has two issues: the observed pairs a_i, q in training data are extremely sparse, meaning that the majority of entries of the interaction matrix cannot be learned effectively. The second problem is that without any constraints on the interaction matrix, this model may overfit due to the large number of parameters. To avoid these two problems, we place low-rank constraints

on the interaction matrix G. The low-rank approximation is widely used in recommender systems [21, 5, 41, 3]:

$$\mathbf{G}^{(i)} \approx \mathbf{Q}^{(i)^T}\mathbf{A} \tag{2}$$

Let k be the dimensionality of the latent factor vectors. $\mathbf{Q}^{(i)} \in \mathbb{R}^{k\times N_q}$ is the latent factor matrix for the queries on position i, $\mathbf{A} \in \mathbb{R}^{k\times N_a}$ is the latent factor matrix for the ads. Plugging Equation 2 back into the interaction model, we get

$$f^{IM}_{q,\mathbf{a}} = \sum_{i=1}^{d} I^{(i)}_{\mathbf{a}} \mathbf{Q}^{(i)^T}_{\cdot q}\mathbf{A}_{\cdot a_i} \tag{3}$$

Like the coefficients in Feature Model, the latent factor vectors of ads and queries $\mathbf{Q}^{(i)}_{\cdot q}$ and $\mathbf{A}_{\cdot a_i}$ could be assumed to be generating from a Gaussian prior,

$$\begin{aligned}\mathbf{Q}^{(i)}_{\cdot q} &\sim \mathcal{N}(0, \lambda_Q^{-1}\mathbf{I}), \ \ i = 1, \cdots, d \\ \mathbf{A}_{\cdot a_i} &\sim \mathcal{N}(0, \lambda_A^{-1}\mathbf{I})\end{aligned} \tag{4}$$

Correlations: However, the above model still fails to capture the connections across positions/ads. To learn the relationships between different positions/ads, we place a matrix-variate normal distribution [17] on $\mathbf{Q} = \left[\mathrm{vec}(\mathbf{Q}^{(1)}), \mathrm{vec}(\mathbf{Q}^{(2)}), \cdots, \mathrm{vec}(\mathbf{Q}^{(d)})\right]$ where $\mathrm{vec}(\cdot)$ denotes the operator which converts a matrix into a vector in a column-wise manner. Then,

$$p(\mathbf{Q}|\mathbf{\Omega}) = \mathcal{MN}_{N_q k\times d}(\mathbf{Q}|\mathbf{Q}', \mathbf{I}_{N_q k} \otimes \mathbf{\Omega})$$

where $\mathbf{Q}'$ is the mean matrix (e.g., zero mean $\mathbf{Q}' = \mathbf{0}_{N_q k\times d}$), $\mathcal{MN}_{u\times v}(\mathbf{X}|\mathbf{M}, \mathbf{U}, \mathbf{V})$ denotes a matrix-variate normal distribution with mean $\mathbf{M} \in \mathbb{R}^{u\times v}$, row covariance matrix $\mathbf{U} \in \mathbb{R}^{u\times u}$ and column covariance matrix $\mathbf{V} \in \mathbb{R}^{v\times v}$. The probability density function of the matrix-variate normal distribution is defined as

$$p(\mathbf{X}|\mathbf{M}, \mathbf{U}, \mathbf{V}) = \frac{\exp(-\frac{1}{2}\mathrm{tr}(\mathbf{U}^{-1}(\mathbf{X}-\mathbf{M})\mathbf{V}^{-1}(\mathbf{X}-\mathbf{M})^T))}{(2\pi)^{uv/2}|\mathbf{U}|^{v/2}|\mathbf{V}|^{u/2}}$$

where $\mathrm{tr}(\cdot)$ and $|\cdot|$ denote the trace and determinant, respectively, of a matrix. More specifically, here the row covariance matrix $\mathbf{I}_{N_q k}$ models the relationships between query latent features, and the column covariance matrix $\mathbf{\Omega}$ models the relationships between different $\mathbf{Q}^{(i)}$'s. In other words, $\mathbf{\Omega}$ models the relationships between positions. We can see that if $\mathbf{Q}' = \mathbf{0}_{N_q k\times d}$ and $\mathbf{\Omega} = \lambda_Q^{-1}\mathbf{I}$, the model $p(\mathbf{Q}|\mathbf{\Omega})$ is equivalent to Eq. 4.

Combined Models (CM): It is straightforward to combine the three models LB, FM and IM. The combined model could be simply:

$$f_{q,\mathbf{a}} = f^{LB}_{q,\mathbf{a}} + f^{FM}_{q,\mathbf{a}} + f^{IM}_{q,\mathbf{a}}$$

Obviously, the combined model is more expressive than any single model. With more flexibility, there is also risk of overfitting using the combined model. Therefore, regularization is more important for the combined model.

4.2 Historical CTR Regularization

Instead of using a zero-mean prior of matrix $\mathbf{Q}$, that is $\mathbf{Q}' = \mathbf{0}_{N_q k\times d}$, we can incorporate historical CTR of pair query q and ad a into the prior information. Here, all positional query latent factors $\mathbf{Q}^{(i)}_{\cdot q}$ are generated from the same corresponding query latent factor $\tilde{\mathbf{Q}}_{\cdot q}$.

$$\mathbf{Q}^{(i)}_{\cdot q} \sim \mathcal{N}(\tilde{\mathbf{Q}}_{\cdot q}, \lambda_Q^{-1}\mathbf{I}), \ \ i = 1, \cdots, d$$

We call $\tilde{\mathbf{Q}}_{\cdot q}$ query latent factors and $\mathbf{Q}^{(i)}_{\cdot q}$ query positional latent factors. Then, we can place the matrix variate gaussian distribution on the $\mathbf{Q}^{(i)}_{\cdot q}$. We duplicate $\mathrm{vec}(\tilde{\mathbf{Q}})$ for d times $\mathbf{Q}' = [\mathrm{vec}(\tilde{\mathbf{Q}}), \cdots, \mathrm{vec}(\tilde{\mathbf{Q}})]$, such that it can be shaped into the same form of $\mathbf{Q}$. Similarly, we can further assume query latent factors $\tilde{\mathbf{Q}}$ is generated from a Gaussian prior. With query latent factors $\tilde{\mathbf{Q}}$ and ad latent factors $\mathbf{A}$, the click-through rate can be incorporated into an optimization framework through traditional collaborative filtering techniques [22],

$$\mathrm{CTR}_{(q,a)} \sim \mathcal{N}(\tilde{\mathbf{Q}}^T_{\cdot q}\mathbf{A}_{\cdot a}, \lambda_c^{-1}\mathbf{I})$$

With the historical CTR regularization, we update the optimization problem as follows,

$$\begin{aligned}\mathrm{O}_c = &\sum_{q,\mathbf{a}} \ell(y_{q,\mathbf{a}}, f_{q,\mathbf{a}}) + \lambda_{\mathbf{b}}\|\mathbf{b}\|_2^2 + \lambda_{\tilde{\mathbf{Q}}}\|\tilde{\mathbf{Q}}\|_2^2 + \lambda_{\mathbf{A}}\|\mathbf{A}\|_2^2 \\ &+kN_q \ln|\mathbf{\Omega}| + \mathrm{tr}((\mathbf{Q}-\mathbf{Q}')\mathbf{\Omega}^{-1}(\mathbf{Q}-\mathbf{Q}')^T) \\ &+\lambda_c \sum_{q,a}(\mathrm{CTR}_{(q,a)} - \tilde{\mathbf{Q}}^T_{\cdot q}\mathbf{A}_{\cdot a})^2 + \mathrm{const}\end{aligned}$$

4.3 Optimization

With the proposed prediction models, we can formalize the click yield prediction problem into an optimization framework. The discrepancy between the estimation $f_{q,\mathbf{a}}$ and the true value $y_{q,\mathbf{a}}$ can be measured by a loss function. We can formalize the problem as an optimization problem as follows

$$\begin{aligned}\mathrm{O}_c = &\sum_{q,\mathbf{a}} \ell(y_{q,\mathbf{a}}, f_{q,\mathbf{a}}) + \lambda_{\mathbf{b}}\|\mathbf{b}\|_2^2 + \lambda_{\mathbf{A}}\|\mathbf{A}\|_2^2 + kN_q \ln|\mathbf{\Omega}| \\ &+\mathrm{tr}(\mathbf{Q}\mathbf{\Omega}^{-1}\mathbf{Q}^T) + \mathrm{const}\end{aligned}$$

The choice of the loss function $\ell(\cdot)$ is also critical to the performance. We will discuss several possible choices for the click yield problem in this section and show the performance comparison of these loss functions on click yield.

4.3.1 *Regression loss:*

One straightforward method is to treat the problem as a regression problem, where we could use the following pointwise loss functions.

Squared error loss (Gaussian): $\ell(y_{q,\mathbf{a}}, f_{q,\mathbf{a}}) = (y_{q,\mathbf{a}} - f_{q,\mathbf{a}})^2$, which is also known as Gaussian response in regression problems.

Huber loss (Huber):

$$\ell(y_{q,\mathbf{a}}, f_{q,\mathbf{a}}) = \begin{cases}\frac{1}{2}(y_{q,\mathbf{a}} - f_{q,\mathbf{a}})^2 & \text{if} |y_{q,\mathbf{a}} - f_{q,\mathbf{a}}| < \delta \\ \delta(|y_{q,\mathbf{a}} - f_{q,\mathbf{a}}| - \delta/2) & \text{otherwise}\end{cases}$$

This function is quadratic for small values of $|y_{q,\mathbf{a}} - f_{q,\mathbf{a}}|$, and linear for large values, with equal values and slopes of the different sections at the two points where $|y_{q,\mathbf{a}} - f_{q,\mathbf{a}}| = \delta$

ϵ-insensitive loss (SVR):

$$\ell(y_{q,\mathbf{a}}, f_{q,\mathbf{a}}) = \begin{cases}0 & \text{if} |y_{q,\mathbf{a}} - f_{q,\mathbf{a}}| < \epsilon \\ |y_{q,\mathbf{a}} - f_{q,\mathbf{a}}| - \epsilon & \text{otherwise}\end{cases}$$

This loss function is used by support vector regression [11]. It has no penalty on any training data whose prediction is close enough to the ground truth (within a threshold ϵ).

4.3.2 *Rank loss*

The problem of click yields prediction is essentially to find the $\mathbf{a}$ which can generate maximum click yields given a query q. From this perspective, learning to rank methods are much preferable to the regression method.

Margin ranking criterion (MRC):

$$\ell^M(\mathbf{a}_1, \mathbf{a}_2) = \sum_{y_{q,\mathbf{a}_1} > y_{q,\mathbf{a}_2}} \max[0, 1 - f_{q,\mathbf{a}_1} + f_{q,\mathbf{a}_2}]$$

MRC considers all pairs of $y_{q,\mathbf{a}_1} > y_{q,\mathbf{a}_2}$, and assign each a cost if the negative label $f_{q,\mathbf{a}_2}$ is larger or within a "margin" of 1. This loss function has a problem that all pairwise violations are considered equally if they have the same margin violation, independent of their position in the list. For this reason the margin ranking loss might not optimize precision at k very accurately.

Weighted Approximately Ranked Pairwise loss (WARP): This loss, proposed in Usunier et al. [34], has been successfully applied in image retrieval tasks [37] and IR tasks [38]. The idea of WARP is to focus more on the top of the ranked list where the top k positions are those we care about, comparing to MRC where no notion of ranked list is introduced. By using the precision at k measure, one can weigh the pairwise violations depending on their position in the ranked list. WARP is defined as an error function as follows:

$$\text{WARP} = \sum_{q,\mathbf{a}} error(f_{q,\mathbf{a}}, y_{q,\mathbf{a}}) = \sum_{q,\mathbf{a}} L(rank(f_{q,\mathbf{a}})) \qquad (5)$$

where $rank(f_{q,\mathbf{a}})$ is the rank of the ads list $\mathbf{a}$, given by $rank(f(q,\mathbf{a})) = \sum_{a' \in \mathcal{S}^-_{q,\mathbf{a}}} \mathbb{I}[f_{q,\mathbf{a}'} > f_{q,\mathbf{a}}]$, where $\mathbb{I}$ is the indicator function, $\mathcal{S}^-_{q,\mathbf{a}} = \{\mathbf{a}' | y_{q,\mathbf{a}'} < y_{q,\mathbf{a}}\}$. Here, rank measure $L(\cdot)$ is the function which transforms the rank to a loss:

$$L(k) = \sum_{j=1}^{k} \alpha_j, \text{with } \alpha_1 \geq \alpha_2 \geq \cdots \geq 0.$$

The idea of the rank function is to compute the violations where negative instances are ranked higher than the positive ones and the L function is to transform the violations into loss. Different choices of α define different importance of the relevance position: for $\alpha_j = 1, \forall i$, we have the same AUC optimization as margin ranking criterion. For $\alpha_j = 1$ and $\alpha_{j>1} = 0$, the precision at 1 is optimized and similarly for $\alpha_{j<N} = 1$ and $\alpha_{j \geq N} = 0$, the precision at N is optimized. For $\alpha_i = 1/i$, a smooth weighting over position is given, where most weight is given to the top position. It is also shown to be superior to other schemes of α and yields state-of-the-art performance [34]. In this work, we will also adopt this setting $\alpha_i = 1/i$.

From Eq. 5, we notice that it is difficult to directly optimize WARP due to the discrete nature of indicator functions. In addition, since the number of negative instances is significantly larger than positive instances, the rank function is inefficient to be calculated. As in [37], Eq. 5 can be readily re-written as

$$\text{WARP} = \sum_{q,\mathbf{a}} \frac{L(rank(f_{q,\mathbf{a}})) \sum_{a' \in \mathcal{S}^-_{q,\mathbf{a}}} \mathbb{I}[f_{q,\mathbf{a}'} > f_{q,\mathbf{a}}]}{rank(f_{q,\mathbf{a}})}$$

with the convention 0/0=0 when the correct label y is top-labeled. WARP can be approximated by the hinge loss $\max(0, 1 - f(q,\mathbf{a}) + f(q,\mathbf{a}'))$ instead of the indicator function, to make the loss function continuous [37]. To perform stochastic gradient descent (SGD) updates, another difficulty is that the rank function is still unknown without computing $f_{q,\mathbf{a}}$ for all $q, \mathbf{a}$. In order to approximate the rank function, for a given pair $q, \mathbf{a}$, one draws negative instances until one is found which violates the indicator function. Thus, the approximate $rank(f_{q,\mathbf{a}})$ by using $\lfloor \frac{D^- - 1}{N} \rfloor$ where $\lfloor \cdot \rfloor$ is the flow function, D^- is the number of items in $\mathcal{S}^-_{q,\mathbf{a}}$ and N is the number of trials of sampling until a violating pair is found. This approximation only requires local knowledge of negative instances, making it possible for SGD style updating rule.

4.3.3 An Efficient Implementation

To solve the problem, we adopt a hybrid optimization approach which mixes stochastic gradient descent and co-ordinate descent. In each iteration, we first perform stochastic gradient descent on $\mathbf{b}, \mathbf{Q}, \mathbf{A}, \tilde{\mathbf{Q}}$. Then, we update $\mathbf{\Omega}$ by the rule $\mathbf{\Omega} \leftarrow \frac{1}{N_q k}(\mathbf{Q} - \mathbf{Q}')^T(\mathbf{Q} - \mathbf{Q}')$ which is obtained by taking the derivative $\mathcal{O}_c$ with respect to $\mathbf{\Omega}$ and setting it to 0. Thus, for each iteration, the time complexity is $O(dkM)$, where d denotes max depth, k denotes the dimensions of latent factors and M is the number of records. In practice, d and k usually are very small (e.g. $d = 4$, $k = 10$), and can be treated as constants. Then, the time complexity for a single iteration will be $O(M)$. Assuming the algorithms need z iterations for convergence, the total time complexity will be $O(zM)$ which allows the algorithms to handle large-scale data.

5. HARVESTING THE TEXTUAL INFORMATION OF ADS

Although the click yield optimization framework proposed in the previous section could utilize many features for the prediction task, there are still issues with leveraging the content features, because the term representation of ad content is extremely sparse. Usually, an ad body only contains around 10 to 20 words. Bag-of-words features (e.g., term frequency or TFIDF) cannot effectively capture the underlying semantics. On the other hand, a latent factor model does not explicitly incorporate ad content, such that it cannot handle cold start ads (the ads do not exist in the training data), which frequently occur in real systems. Topic models [2, 45, 19] have been developed to learn the latent semantics of texts. A number of approaches, such as LDA [2] and sparse topical coding [45], could be used in our problem. In this work, we adopt the topical coding model [45] to incorporate ad content into our click yield optimization framework. The reasons for using topical coding are as follows: 1) the topical coding model is a non-probabilistic formulation of topic models for discovering latent representations of large collections of data. It can easily fit into our optimization framework. 2) it can directly control the sparsity of inferred representation by using an appropriate regularizer. 3) the integrated model can be solved efficiently. We treat the content of each ad as a document. For simplicity, we use a as the index of an ad. Let $V = \{1, ..., N_v\}$ be the vocabulary with N_v words. Then w_{an} represents the raw word count of term n in ad a. Let $\theta_a \in \mathbb{R}^k$ represent the document code, playing a similar role as $P(z|\theta)$ in traditional topic models like probabilistic latent semantic analysis (PLSA) [19] or latent Dirichlet allocation (LDA) [2]. Similarly, let $\beta \in \mathbb{R}^{k \times N}$ be a dictionary with k bases, which can be treated as $P(z|w)$ in LDA or PLSA.

For θ, due to the connection with the click yield model, we can place either Gaussian prior $p(\theta) \propto \exp(-\lambda_\theta \|\theta\|_2^2)$ or Laplace prior $p(\theta) \propto \exp(-\lambda_\theta \|\theta\|_1)$ for L_2 and L_1 regularization respectively rather than Dirichlet prior. Then let $\mathbf{\Theta} = \{\theta_a\}_{a=1}^{N_a}$ denote the codes for a collection of ads $\{\mathbf{w}_a\}_{a=1}^{N_a}$. We learn the parameter by maximimum a posteriori (MAP) estimation. Unlike [45], since we only use ad level topical coding in our methods, we remove word level coding. Then we have

$$\mathcal{O}_t = \min_{\boldsymbol{\theta}, \beta} \sum_{a,n} -\log Poiss(w_{an}; \theta_a^T \beta_{\cdot n} + \delta \cdot \beta_n^b) + \lambda_\theta \sum_d \|\boldsymbol{\theta}_d\|_2^2$$
$$\text{s.t. } : \theta_a \geq 0, \forall a; \beta_k \in \mathcal{P}, \forall k, \delta > 0, \beta^b \in \mathcal{P},$$

where $\ell(\theta_a, \beta)$ is Poisson loss, and alternatively we can place a background topic β_n^b.

$$\begin{aligned}\ell(\theta_a, \beta) &= -\log Poiss(w_{an}; \theta_a^T \beta_{.n} + \delta \cdot \beta_n^b) \\ &= -w_{an} \log(\theta_a^T \beta_{.n} + \delta \cdot \beta_n^b) + \theta_a^T \beta_{.n} + \delta \cdot \beta_n^b\end{aligned}$$

The optimization problem can be solved efficiently due to three facts: 1) the property of multilinearity [29], which means that the model is linear with respect to each model parameter when others are fixed, 2) Proposition 1 in [45] states that the optimal value of a single parameter when others are fixed is the maximum between zero and the value obtained by a non-constrained version of the same problem. 3) Efficient methods [12] exist to project real-valued vectors onto the simplex.

To incorporate the topical coding into our click yields model, we explore two methods: one through features and the other through latent factors.

5.1 Topical Coding Through Features

In this form, we put the topical coding of ads into ad features and the new ad features become $\tilde{\mathbf{x}}_a = [\mathbf{x}_a^T, \theta_a^T]^T$. With the shared model parameter θ, we connect the two optimization problem $\mathcal{O}_c$ and $\mathcal{O}_t$. This form is similar to the supervised topic model [1] and MedLDA [44]. The final optimization problem is

$$\mathcal{O}_c + \lambda \cdot \mathcal{O}_t$$

where λ is a weight parameter for $\mathcal{O}_t$. Then, we perform stochastic gradient descent to update θ, referring to the total optimization.

When passing a sample of click $y_{q,\mathbf{a}}$ and $a \in \mathbf{a}$, we have the following updates

$$\theta_a \leftarrow \mathcal{P}\left\{\theta_a - \eta \frac{\partial \mathcal{O}_c(y_{q,\mathbf{a}})}{\partial \theta_a}\right\}$$

Similarly, when passing a sample of word count w_{an}, we have

$$\theta_a \leftarrow \mathcal{P}\left\{\theta_a - \eta \frac{\partial \mathcal{O}_t(w_{an})}{\partial \theta_a}\right\}$$

where $\mathcal{P}\{\cdot\}$ is the projection function which truncates θ_a and guarantees $\theta_a \geq 0$

5.2 Topical Coding Through Latent Factors

An alternative method that can incorporate topical coding of ads is through latent factors. Without constraints, the latent representations of an ad learned from $\mathcal{O}_c$ and $\mathcal{O}_t$ are different. To connect the two problems, we can place constraints on the two latent representations of ads, such that the information of ad content can be incorporated into click yields optimization. One natural approach to require the two latent factors be the same:

$$\theta_a = \mathbf{A}_{.a}, \forall a$$

In this method, the objective function will be $\mathcal{O}_c + \lambda \cdot \mathcal{O}_t$ that is the same as the one in Sec. 5.1 and similar optimization methods (e.g., stochastic gradient descent) can be used to solve the problem. Another more flexible approach is to use a regularizer to keep the two latent factors similar rather than identical, as:

$$\lambda_{\theta\mathbf{A}} \|\theta_a - \mathbf{A}_{.a}\|_2^2, \forall a$$

The objective function will be $\mathcal{O}_c + \lambda_{\theta\mathbf{A}} \sum_a \|\theta_a - \mathbf{A}_{.a}\|_2^2 + \lambda \cdot \mathcal{O}_t$. We can also view one latent factor as a sample drawing from a multivariate normal distribution with the mean of the other latent factor: $\mathbf{A}_{.a} \sim \mathcal{N}(\theta_a, \lambda_{\theta\mathbf{A}}^{-1}\mathbf{I})$. This will produce the formalism in Wang and Blei's work [35]. Also, the SGD-style optimization can be used to solve the problem. In our studies, we find the performances of the two methods are comparable and we will use the first one in the following experiments.

Table 2: Data sets

	April 2012	May 2012	total
Impressions	29,722,684	13,160,289	42,882,973
$(q, \mathbf{a})$	112,084	52,814	135,445
Ads	62,423	34,214	72,959
Queries	49,483	26,104	53,730

6. EXPERIMENTS

In this section, we analyze variations of the proposed model and compare our model with the state-of-the-art methods.

6.1 Experiment Setting

We collected the search log data from a commercial search engine in the U.S. market in April 2012 and the first two weeks of May 2012. The dataset in April 2012 is the same one as we used in Section 3. We filter out the low frequency queries and ads to remove the noise. In this work, we focus on head queries for three reasons: 1) Head queries hold the majority of the search traffic which is more important to the search engine. 2) Due to high traffic of head queries, user behaviors are usually consistent and we can mine the patterns of user behaviors effectively. 3) In this work, since we investigate ad list effects rather than individual ads, we do not have enough data to analyze tail queries. Finally, the April 2012 data set contains 29,722,684 impressions and the May 2012 data set contains 13,160,289 impressions. The other data statistics are shown in Table 2. We used April 2012 dataset as training set, and the May 2012 dataset as the test set. Since the train/test data are split temporally rather than randomly, we avoid the problem of predicting the past using future data. This evaluation is more consistent with a real application scenario. At the same time, the cold start problem becomes serious.

To evaluate the performance of click yields prediction, we report the standard ranking measurements—Mean Average Precision (MAP) and Precision@N for references.

By using our click yield prediction framework, with an issued query, we can reselect the group of ads to display with the higher predicted click yields, and it will improve the performance of current search engine. Solving this mathematical problem in runtime may not be affordable when the number of ad candidates is large. Therefore, we can only do it for the popular queries in off-line mode. We argue that these queries take the majority of the ad clicks in the traffics. To evaluate this relative improvement, we present a novel evaluation metric, referred to as Relative Gain of Click Yields (RGCY). The overall click yield of the current system is $\underline{\text{CY}} = \frac{1}{\text{Imp}} \sum_q \sum_{\mathbf{a}} y_{q,\mathbf{a}} \cdot \text{Imp}(q, \mathbf{a})$. Given a query q, ideally, if we know the optimal $\mathbf{a}$, which can generate the maximum click yield, that is for all impressions of q, one can always show this optimal result $\mathbf{a}_{opt(q)}$ which is defined $\mathbf{a}_{opt(q)} \in \{\mathbf{a}' | \forall \mathbf{a}, y_{q,\mathbf{a}'} \geq y_{q,\mathbf{a}}\}$. The maximum click yield which the system could have is $\overline{\text{CY}} = \frac{1}{\text{Imp}} \sum_q \tilde{y}_q \sum_{\mathbf{a}} \text{Imp}(q, \mathbf{a})$ where $\tilde{y}_q = y_{q,\mathbf{a}_{opt(q)}}$ denotes the optimal click yield for query q.

Given a query q and its ranked ad lists $l_q = \{\mathbf{a}_{(1)}, \mathbf{a}_{(2)}, \cdots, \mathbf{a}_{(n)}\}$ where $f_{q,\mathbf{a}_{(1)}} \geq f_{q,\mathbf{a}_{(2)}} \cdots \geq f_{q,\mathbf{a}_{(n)}}$, $\text{CY@}N = \frac{1}{\text{Imp}} \sum_q \hat{y}_q \sum_{\mathbf{a}} \text{Imp}(q, \mathbf{a})$ where $\hat{y}_q$ is the average click yield of top N ad lists for query q, $\hat{y}_q = \frac{1}{N} \sum_{i=0}^{N} y_{q,\mathbf{a}_{(i)}}$. We

define the relative gain of click yield to be

$$\text{RGCY@}N = \frac{\text{CY@}N - \underline{\text{CY}}}{\overline{\text{CY}} - \underline{\text{CY}}}$$

It is trivial to see that $\overline{\text{CY}} \geq \text{CY@}N \geq \underline{\text{CY}}$. The RGCY@$N$ actually measures the gap with the maximum click yield. More specifically, RGCY@1 is more critical to the system since it can generate the maximum click yield.

6.2 Model Analysis

In this section, we detail the anatomy of the proposed model and systematically analyze the contributions and effects of each part.

Estimation Model Analysis: We first test the scoring model. The results are shown in Figure 3(a). In this experiment, the performance is evaluated by PREC@1 and MAP, and the loss function is WARP. We can see that the performances are consistent in terms of PREC@1 and MAP. The bias model performs the worst on the left. With the feature model added, the model performs better. When we further incorporate the interaction model into the system, the model achieve the best performance in both PREC@1 and MAP.

Convergence Check: We also show the learning curve of the combined in Figure 3(b), where two loss functions are tested: Gaussian response and WARP. The convergence of the two loss functions can be achieved after 100 iterations. We can also see that the performance of the ranking loss (WARP) is much better than the regression loss (Gaussian response).

Position Correlation Analysis: Matrix $\boldsymbol{\Omega}$ is shown in Figure 3(c). We can see from the figure that position 1 is positively correlated to the position 4, while position 1 is negatively correlated to position 2 and position 3. This results are consistent with our intuitions: if the ads in position 2 and position 3 are similar to the ads in the top position, users are more likely to skip the ads due to information redundancy, which may decrease click yields. On the other hand, the ads at the bottom position (position 4) are positively correlated with position 1. One possible reason is that position 4 is far apart from the top position, and similarity between them is insensitive such that the click yields might increase.

Loss Function: From Figure 3(b), we have already seen the performance differences between different loss functions (WARP and Gaussian). In this experiment, all five loss functions in Section 4.3 are systematically compared. The results are shown in Table 3. For pointwise regression loss, we can see that the least square loss and ϵ-insensitive loss (SVR) can generate comparable performance. Huber loss, which is quadratic for small errors and linear for large errors, can achieve slightly better performance than the two other pointwise regression losses in this problem. On the other hand, for pairwise ranking loss, WARP can be treated as a weighted version of MRC, which emphasizes the top positions weight. From Table 3, we can see that the overall performance (MAP) of WARP is similar to MRC, while the performance of WARP on the RGCY@1 which only evaluates the accuracy of top position (RGCY@1 can be considered as the weighted version of MRC) is much better than MRC. Overall, comparing with the pointwise regression losses, the two pairwise rank losses are preferable in click yield prediction in all three evaluations (PREC@1, MAP and RGCY@1).

6.3 Side Information

For the topical coding model, we explore how topics are learned. The dictionary $\boldsymbol{\beta}$ can be interpreted as a topic matrix as in standard topic models. We can describe topics as in other topic models by ranking terms in probabilities. We show some example topics in Table 4. We can see that these topics can be easily recognized. Moreover, modeling ad content is not only useful for explanatory analysis, it indeed improves the prediction tasks. From Table 3, we can see that by incorporating the topical coding of ads, we get further improvement on RGCY@1.

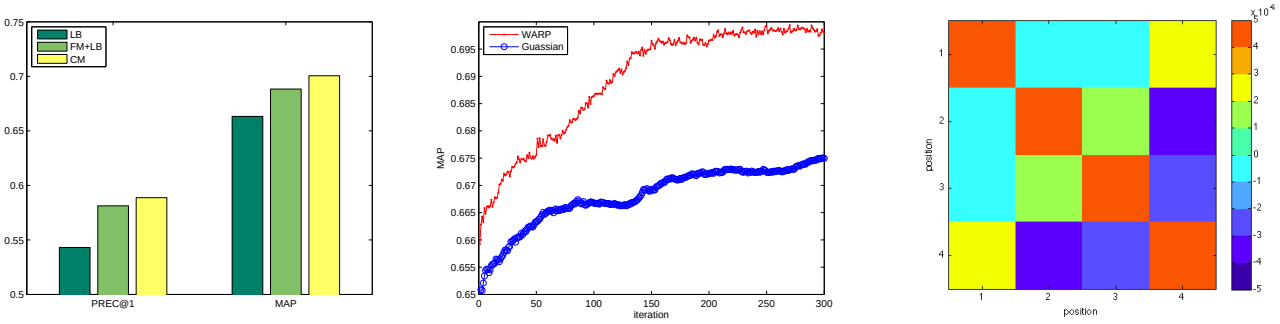
Next, we examine the effects of two side information: historical CTR regularization and topical modeling of ad content. The results are shown in Table 3, where CTR represents historical CTR regularization, TCF represents topical model incorporated through features, TCL represents topical model incorporated through latent factors. We see that the performances improve slightly across all loss functions when adding historical CTR regularization into the system. When adding topical information, we see that the performance becomes much better than the original model, due to alleviating the cold-start and sparseness problems. We notice that WARP loss with CTR regularization and incorporated topical coding through latent factors achieve relative better performance than other combinations (although MRC is slightly better than WARP for MAP under CTR,TCL). In the following section, we will use it to compare with the other existing methods.

Table 3: Predictive results of Click Prediction

Side	loss	PREC@1	MAP	RGCY@1
N/A	Gaussian	0.5683	0.6792	0.6475
CTR	Gaussian	0.5689	0.6796	0.6510
CTR,TCF	Gaussian	0.5789	0.6894	0.6592
CTR,TCL	Gaussian	0.5771	0.6887	0.6701
N/A	Huber	0.5694	0.6811	0.6617
CTR	Huber	0.5695	0.6811	0.6618
CTR,TCF	Huber	0.5714	0.6834	0.6627
CTR,TCL	Huber	0.5730	0.6854	0.6679
N/A	ϵ-Insensitive	0.5785	0.6886	0.6365
CTR	ϵ-Insensitive	0.5787	0.6889	0.6401
CTR,TCF	ϵ-Insensitive	0.5824	0.6925	0.6402
CTR,TCL	ϵ-Insensitive	0.5833	0.6941	0.6416
N/A	MRC	0.5839	0.6930	0.6641
CTR	MRC	0.5848	0.6932	0.6661
CTR,TCF	MRC	0.5849	0.6936	0.6697
CTR,TCL	MRC	0.5859	0.6946	0.6732
N/A	WARP	0.5841	0.6923	0.6685
CTR	WARP	0.5845	0.6924	0.6782
CTR,TCF	WARP	0.5851	0.6936	0.6829
CTR,TCL	WARP	0.5859	0.6943	0.6882

Table 4: Examples of topics are shown. The terms are top ranked terms in each topic. The topic names shown in bold are given by the authors

Shopping	**Video**	**Travel**	**Finance**	**Game**
low	tv	book	online	games
free	order	hotels	credit	play
order	free	rates	apply	store
shop	time	price	card	online
shipping	watch	insurance	get	free
quality	college	car	university	today
today	live	parts	gift	guarantee
orders	24	hotel	degree	official
high	take	great	website	fun
prices	like	best	back	enjoy
supplies	sears	auto	earn	favorite
site	movies	low	college	kids
state	community	quotes	magazine	better
official	episodes	deals	cash	toys



(a) Model Analysis (b) WARP vs. Gaussian (c) Co-relation across positions

Figure 3: Experimental results for model analysis

6.4 Comparison with existing methods

In this section, we name our methods CYP for Click Yield Prediction. Although there is no previous work on the click yield prediction problem, some models could be easily adapted to solve the problem. We compare our model with four existing methods:

- **Probabilistic Matrix Factorization (PMF)** is a popular method [31] in collaborative filtering. Here, we treat queries as the users and ads as the items to be recommended for users. CTR is the response in this framework.
- **User Browsing Model (UBM)** is the baseline method and a classical click model [13]. We adopt the EM inference process for parameter α and β. To avoid infinite values in log-likelihood, α is truncated into the range $(0.0001, 0.9999)$.
- **Matrix Factorization Click Model (MFCM)** is proposed by Shen et al. [32]. The model incorporates collaborative filtering techniques into the click model.
- **Relational click model (CRF)** proposed by Xiong et al. [39]. They adopt Conditional Random Fields (CRF) to model the exclusive effects between similar ads which are shown together in the same impression.

We choose these four methods to compare to various alternative methods to solve this problem. PMF is a classical recommender method, UBM is a classical click model while CFCM and CRF are very recent methods and the state-of-the-arts in modeling clicks. All four methods follows the classical two-step approach, that is, predicting CTR of individual ad at first and then compose the click yields for ad lists and rank them. The comparison results (on precision, RGCY and MAP) are shown in Table 5. We notice that the performances across five evaluations are consistent: the two classical methods—PMF and UBM—show similar performance on this problem, and they are relatively worse than the two recent click models—MFCM, CRF. MFCM which incorporates examination hypothesis into matrix factorization, can get slightly better results than matrix factorization. CRF is the first click model which tries to model the relational ad clicks on the ads displayed together. The performance of CRF is much better than other three baselines which only focus on modeling clicks on single ad. However, for the click yield prediction problem, because all these four methods are two-step approaches and not designed to optimize click yield directly, our method outperforms all these methods noticeably. Especially, on RGCY@1, our methods can achieve 0.6882 which almost improves the click yield 10% absolutely, comparing to 0.5975—the value of the second best method, CRF.

Table 5: Comparison with existing methods

Model	PREC@1	PREC@5	MAP	RGCY@1	RGCY@5
CYP	**0.5859**	**0.7428**	**0.6943**	**0.6882**	**0.7438**
PMF	0.5251	0.6828	0.6450	0.4991	0.5597
UBM	0.5338	0.6845	0.6647	0.4519	0.5015
MFCM	0.5553	0.6985	0.6754	0.5830	0.6293
CRF	0.5760	0.7320	0.6887	0.5975	0.7175

7. CONCLUSION

In this paper, we have studied ad group performance—click yield, which is a important measurement for ad delivery. Compared to traditional methods, we provide a new perspective to measure group performance of ads displayed together. We systematically explored different aspects: bias model, features, interactive influence, depth, and correlation across positions. Additionally, to best leverage the text features and solve the sparseness issue in textual information and cold starts on ads, we incorporate a topic coding model into our framework to learn the topic information of short texts for ads in two ways—through features and through latent factors. Finally, various loss functions are also studied. We find that ranking loss is preferred for this problem.

We collected a large-scale real world dataset from a commercial search engine to conduct experiments. Our experiments show that directly predicting click yields achieves noticeable improvement compared to state-of-the-art two-step approaches.

Several interesting points remain to be explored in the future: first, research beyond the prediction of group performance in sponsored search would be valuable. A more interesting problem is, given a query, find the best depth and the ad list that can generate the best click yields for search engines. This is a much harder problem, similar to a combinatorial optimization problem. Second, although we mainly study top queries in this work, studying tail queries is also interesting and necessary as future work.

8. REFERENCES

[1] D. M. Blei and J. D. McAuliffe. Supervised topic models. In *NIPS*, 2007.

[2] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, January 2003.

[3] G. Bouchard, D. Yin, and S. Guo. Convex collective matrix factorization. In *Proceedings of AISTATS 2013*.

[4] A. Broder and V. Josifovski. Introduction to computational advertising, 2011. Retrieved from http://www.stanford.edu/class/msande239/.

[5] B. Cao, D. Shen, K. Wang, and Q. Yang. Clickthrough log analysis by collaborative ranking. In *AAAI*, 2010.

[6] O. Chapelle and Y. Zhang. A dynamic bayesian network click model for web search ranking. In *WWW '09*.

[7] D. Chen, W. Chen, H. Wang, Z. Chen, and Q. Yang. Beyond ten blue links: enabling user click modeling in federated web search. In *Proceedings of WSDM '12*, 2012.

[8] Y. Chen and T. W. Yan. Position-normalized click prediction in search advertising. In *Proceedings of KDD '12*.

[9] H. Cheng and E. Cantú-Paz. Personalized click prediction in sponsored search. In *Proceedings of WSDM '10*.

[10] N. Craswell, O. Zoeter, M. Taylor, and B. Ramsey. An experimental comparison of click position-bias models. In *WSDM '08*.

[11] H. Drucker, C. J. C. Burges, L. Kaufman, A. J. Smola, and V. Vapnik. Support vector regression machines. In *NIPS*, pages 155–161, 1996.

[12] J. Duchi, S. Shalev-Shwartz, Y. Singer, and T. Chandra. Efficient projections onto the l1-ball for learning in high dimensions. In *Proceedings of ICML '08*, 2008.

[13] G. E. Dupret and B. Piwowarski. A user browsing model to predict search engine click data from past observations. In *SIGIR '08*.

[14] T. Graepel, J. Q. Candela, T. Borchert, and R. Herbrich. Web-scale bayesian click-through rate prediction for sponsored search advertising in microsoft's bing search engine. In *ICML '10*.

[15] F. Guo, C. Liu, A. Kannan, T. Minka, M. Taylor, Y.-M. Wang, and C. Faloutsos. Click chain model in web search. In *WWW '09*.

[16] F. Guo, C. Liu, and Y. M. Wang. Efficient multiple-click models in web search. In *Proceedings of WSDM '09*, 2009.

[17] A. K. Gupta and D. K. Nagar. *Matrix Variate Distributions*. Chapman & Hall, 2000.

[18] D. Hillard, S. Schroedl, E. Manavoglu, H. Raghavan, and C. Leggetter. Improving ad relevance in sponsored search. In *Proceedings of WSDM '10*

[19] T. Hofmann. Unsupervised learning by probabilistic latent semantic analysis. *Machine Learning*, 42(1-2):177–196, January-February 2001.

[20] T. Joachims, L. Granka, B. Pan, H. Hembrooke, and G. Gay. Accurately interpreting click through data as implicit feedback. In *Proceedings of SIGIR '05*, 2005.

[21] Y. Koren. Factor in the neighbors: Scalable and accurate collaborative filtering. *ACM Transactions on Knowledge Discovery from Data*, 4(1):1–24, January 2010.

[22] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

[23] C. Liu, F. Guo, and C. Faloutsos. Bbm: bayesian browsing model from petabyte-scale data. In *KDD '09*, 2009.

[24] C. Liu, F. Guo, and C. Faloutsos. Bayesian browsing model: Exact inference of document relevancfe from petabyte-scale data. *ACM Trans. Knowl. Discov. Data*, 4(4):19:1–19:26, Oct. 2010.

[25] T.-Y. Liu. Learning to rank for information retrieval. *Found. Trends Inf. Retr.*, 3(3):225–331, Mar. 2009.

[26] A. K. Menon, K.-P. Chitrapura, S. Garg, D. Agarwal, and N. Kota. Response prediction using collaborative filtering with hierarchies and side-information. In *KDD '11*.

[27] T. Minka, J. Winn, J. Guiver, and A. Kannan. A click through model - sample code. `http://research.microsoft.com/en-us/um/cambridge/projects/infernet/docs/Click%20through%20model%20sample.aspx`, 2009.

[28] T. Moon, A. Smola, Y. Chang, and Z. Zheng. Intervalrank: isotonic regression with listwise and pairwise constraints. In *WSDM '10*.

[29] S. Rendle. Factorization machines with libFM. *ACM Trans. Intell. Syst. Technol.*, 3(3):57:1–57:22, May 2012.

[30] M. Richardson, E. Dominowska, and R. Ragno. Predicting clicks: estimating the click-through rate for new ads. In *WWW '07*.

[31] R. Salakhutdinov and A. Mnih. Probabilistic matrix factorization. In *NIPS 21*, pages 1257–1264. 2008.

[32] S. Shen, B. Hu, W. Chen, and Q. Yang. Personalized click model through collaborative filtering. In *WSDM '12*.

[33] R. Srikant, S. Basu, N. Wang, and D. Pregibon. User browsing models: relevance versus examination. In *KDD '10*.

[34] N. Usunier, D. Buffoni, and P. Gallinari. Ranking with ordered weighted pairwise classification. In *ICML '09*.

[35] C. Wang and D. M. Blei. Collaborative topic modeling for recommending scientific articles. In *KDD '11*, 2011.

[36] X. Wang and et al. Click-through prediction for sponsored search advertising with hybrid models. In *KDDCUP 2012*, 2012.

[37] J. Weston, S. Bengio, and N. Usunier. Large scale image annotation: learning to rank with joint word-image embeddings. *Mach. Learn.*, 81(1):21–35, Oct. 2010.

[38] J. Weston, C. Wang, R. Weiss, and A. Berenzweig. Latent collaborative retrieval. In *ICML*, 2012.

[39] C. Xiong, T. Wang, W. Ding, Y. Shen, and T.-Y. Liu. Relational click prediction for sponsored search. In *Proceedings of WSDM '12*, 2012.

[40] W. Xu, E. Manavoglu, and E. Cantu-Paz. Temporal click model for sponsored search. In *SIGIR '10*, 2010.

[41] D. Yin, S. Guo, B. Chidlovskii, B. D. Davison, C. Archambeau, and G. Bouchard. Connecting comments and tags: Improved modeling of social tagging systems. In *Proceedings of WSDM '13*.

[42] W. V. Zhang and R. Jones. Comparing click logs and editorial labels for training query rewriting. In *Query Log Analysis: Social And Technological Challenges. A workshop at WWW 2007*.

[43] Y. Zhang, W. Chen, D. Wang, and Q. Yang. User-click modeling for understanding and predicting search-behavior. In *KDD '11*.

[44] J. Zhu, A. Ahmed, and E. P. Xing. Medlda: maximum margin supervised topic models for regression and classification. In *Proceedings of ICML '09*, 2009.

[45] J. Zhu and E. Xing. Sparse topical coding. In *UAI '11*, 2011.

[46] Z. A. Zhu, W. Chen, T. Minka, C. Zhu, and Z. Chen. A novel click model and its applications to online advertising. In *WSDM '10*, 2010.