Improving Relevance Prediction by Addressing Biases and Sparsity in Web Search Click Data

[*] Qi Guo, Dmitry Lagun, Denis Savenkov, Qiaoling Liu
Mathematics & Computer Science
Emory University
{qguo3,dlagun,denis.savenkov,qiaoling.liu}@emory.edu

ABSTRACT

In this paper, we present our approach and findings in participating the 2012 Yandex Relevance Prediction Challenge. Our approach has two goals: on one hand, we aim to address four types of biases, namely, position-bias, perception-bias, query-bias, and session-bias to better interpret the clickthrough information; on the other hand, we aim to address the clickthrough sparsity by exploiting various back-off strategies. We use gradient boosted regression trees to combine the different features and model the interactions among them. Our final submission ranks 3rd (AUC 0.6635) among the prize eligible participants on the first subset of test queries, but drops to 8th (AUC 0.6536) on the second (hidden) subset, which is potentially due to over-fitting. In this paper, we also discuss our post-competition efforts in addressing this issue through cross-validation and more careful model selection.

Categories and Subject Descriptors

H.3.3 [**Information Systems**]: Information Storage and Retrieval

General Terms

Design, Experimentation, Human Factors

Keywords

relevance prediction, user behavior modeling, biases and sparsity

1. INTRODUCTION

Predicting the relevance of URLs based on user search behavior such as clickthrough is an essential yet very challenging problem. The 2012 Yandex Relevance Prediction Challenge was held to consolidate and scrutinize the work on this problem, by providing a fully anonymized dataset shared by Yandex which has clicks information and relevance judgements. In this paper, we present our approach and findings in participating in the Relevance Prediction Challenge. Clickthrough has been shown valuable for inferring search result relevance, but the usefulness of clickthrough data is limited by a number of biases and its sparsity. Some biases that strongly influence the clickthrough are:

- Position-bias: Search results are presented as a ranked list, lower-ranked URL are less likely to be examined. As a result, a lower-ranked relevant URL may have lower CTR (clickthrough rate) than a higher-ranked irrelevant URL simply because very few searchers have seen the URL.
- Perception-bias: a searcher's action is based primarily on the "perceived" relevance, where a searcher guesses the URL relevance based on a short summary generated by the search engine. However, the "perceived" relevance may be inconsistent with the actual "intrinsic" relevance, where a searcher clicks on a result may end up finding that it is not relevant.
- Query-bias: User behavior varies for different queries. For example, 20 seconds of dwell time may indicate relevance for an easy query but irrelevance for a difficult one.
- Session-bias: The session-bias includes both behavioral variations from user and intent. Note that, intent is different from query as a query may carry multiple intents while an intent can be represented by different queries.

In addition to the above biases, clickthrough also suffers from sparsity – it is a good indicator of document relevance for relatively frequent queries, but for infrequent queries/URLs clickthrough tends to be sparse and therefore not very reliable. To improve the relevance prediction from mining the searcher interaction data, our approach aims at addressing the above issues. The rest of paper is organized as follows. We first overview the related work in Section 2 and describe the task of the challenge in more detail in Section 3. Then, in Section 4, 5, and 6, we describe the features, learning algorithms, and model selection we used for tackling the challenge. We then present and discuss the results and findings in Section 7. Finally, Section 8 concludes the paper.

2. RELATED WORK

In this section, we will describe the related work in addressing the biases and sparsity issues in user behavior. To address the position bias, several click models are proposed such as the Cascade model [7], the user browsing model (UBM) [9], the Dynamic Bayesian Network (DBN) click model, the Dependent Click Model (DCM) [12], and the Click Chain Model (CCM) [11]. Our approach adapts the DBN model and incorporates the predicted relevance as part of our feature set. Many other models are proposed to distinguish the "perceived relevance" and "intrinsic" relevance, such as the Session Utility Model (SUM) [8] and the Post-Click

[*] The first three authors contributed equally to this work

WSCD 2012, February 12, 2012, Seattle WA, USA
Copyright 20XX ACM X-XXXXX-XX-X/XX/XX ...$10.00.

Click model (PCC) [21]. For example, Dupret and Liao [8] assumed that all sessions ending with a click are successful, and the last clicked document must contribute to the success. Zhong et. al. [21] combined post-click behaviors (e.g. the page dwell time) with click behaviors to provide an unbiased estimation of relevance. The usefulness of the post-click browsing behavior for improving web search ranking was also demonstrated by Agichtein et. al. [2]. Inspired by this prior work, we also derive features based on the page dwell time and last clicks of queries for addressing the perception bias. A well-known classification of web queries proposed by Broder [4] is into three types: navigational, informational, and transactional. Different types of queries could lead to quite different user behavior, e.g. for navigational queries, users leave immediately after reaching the target site; while for informational queries, users may need to click several pages to acquire the information. In a slightly different dimension, researchers [3, 1, 13] also found that that user behavior varies as the level of query difficulty changes – for example, searchers working on difficult queries tend to spend a longer time on the search result page than on easy queries. Considering these differences, we compute query-level features and their deviations of URL features to alleviate the query bias. To characterize the bias between the user search intent and the query in a search session, Hu et. al. [14] proposed the intent hypothesis. Zhang et. al. [20] further proposed a Task-Centric click Model (TCM) by characterizing user behavior in a search session as an entirety. The model is demonstrated to be especially useful for infrequent queries. Based on the intent hypothesis, we compute session-level features with normalization to account the session-bias. As many queries and documents have no or very few clicks, the usefulness of using clickthrough features for predicting relevance is very limited. Therefore, Craswell and Szummer [6] proposed to use random walks on the click graph to smooth the noisy and sparse click data. Gao et. al. [10] compared two other smoothing methods: query clustering and discounting. Besides using the smoothing techniques, in this paper we also proposed various back-off strategies to solve the sparsity problem.

3. PROBLEM STATEMENT

The task is to predict labels of documents for the given test set of queries, using the shared dataset containing the search log and queries with labeled URLs. The search log is supposed to be used both for training the prediction models and for prediction of the labels for the test set of queries.

3.1 Dataset

The dataset includes user sessions extracted from Yandex logs, with queries, URL rankings and clicks. Unlike previous click datasets, it also includes relevance judgments for the ranked URLs, for the purposes of training relevance prediction models. To allay privacy concerns the user data is fully anonymized. So, only meaningless numeric IDs of queries, sessions, and URLs are released. The queries are grouped only by sessions and no user IDs are provided. The dataset consists of several parts. Specifically, in the dataset, there are 30,717,251 unique queries, 117,093,258 unique urls, 43,977,859 sessions, and 340,796,067 records in total. 71,930 query-region-url triples for the total query set (training + test) were assessed and 8,410 query-region pairs were with assesed urls (training + test). The logs are about two years old and do not contain queries with commercial intent detected with Yandex proprietary classifier.

Relevance Labels: Labels were assigned by Yandex judges to a subset of URLs appearing in the logs. Labels are binary: Relevant (1) and Irrelevant (0). Labels were assigned during the course of a year after the logs had been collected. URLs were judged based not only on the text of the query, but also on the region of the user, if it was necessary, but not in every case. So, the presence of the RegionID does not necessarily mean that the relevance is region-specific.

3.2 Evaluation metric

Submissions were evaluated using Area Under receiver operating Curve (AUC) [15] metric, which was calculated using the ranking of URLIDs provided by participants for each query and then averaged across all queries. Only judged documents were considered, i.e. all documents without labels were ignored during the evaluation. AUC has been used as measure of ranking performance of binary classifiers and has been shown to be more robust than accuracy metric [15]. The AUC represents the probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one. In terms of pairwise comparison AUC can be calculated as ratio of the number of correct pairwise orderings to the number of all possible pairs. This quantity is also called Wilcoxon-Mann-Whitney (WMW) [19] and can be expressed as follows:

$$W = \frac{\sum_{i=0}^{p-1} \sum_{j=0}^{n-1} I(x_i, y_j)}{pn} \tag{1}$$

$$I(x_i, y_j) = \begin{cases} 1 & \text{if } f(x_i) > f(y_j) \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

where p is the number of relevant document, n is the number of not relevant documents, x_i is the i^{th} relevant document and y_j is the j^{th} not relevant document. The function $f(x)$ assigns the score to the document x which is used to derive document ranking.

4. FEATURES

Clickthrough rate (CTR) is a good indicator of document relevance as it is a signal about user preferences among presented search results. However, the four types of biases and data sparsity make it hard to predict document relevances based on raw CTR. In this section, we will describe the features we develop for addressing the four types of biases and data sparsity.

Position-Bias: The raw CTR calculation assumes that all the shown documents are examined, which is not always the case, especially for low-ranked documents. To this end, we adapt the Dynamic Bayesian Network (DBN) click model. In addition, we calculated CTR for each position separately, that is clicks when document was on i-th position divided by the number of impressions on i-th position. This could help machine learning algorithm to find separate CTR thresholds for different positions. [17] suggests that "Click-Skip above" and similar behavior patterns can be a good evidence of pairwise document preferences. To get use of this idea we included the following features: the number of impressions when document below was clicked but current document wasn't clicked divided by the total number of impressions, clicks when document above was skipped divided by the number of clicks and a couple of similar features (document clicked, but both documents were not around and vice versa).

Perception-Bias: As clickthrough reflects "perceived relevance", we exploit post-click signals such as dwell time and session-level click sequence to model "intrinsic relevance". For example, longer dwell time and the last click in a query/session may indicate higher relevance. Along with average/median dwell time per click, and the number of times the document was clicked last in the session we

Name	*Description*	*Group*
UrlTimeBeforeClk	average time before url is clicked in SERP	-
MultClkShows	query shows with more then one click on the given url	-
OnlyClkInSession	number of times the clicked url was the only click in session	-
ithPosCtr	url CTR on i-th position	position-bias
DBN	url relevance by DBN model	position-bias
LastClkInSerp	the fraction of clicks which were the last for query	position-bias
SkipClkAboveBelow	url skipped, clicked below and above impressions	position-bias
SkipClkBelow	url skipped, clicked below impressions	position-bias
ClkSkipAbove	url clicked, skipped above impressions	position-bias
InverClicks	url inversion clicks rate (click below, then click on the given url)	position-bias
AveClkPos	average click rank for url	position-bias
DwellPerClk	url average dwell time per click	perception-bias
SatClk/TotalClk	the number of Sat clicks over total number of clicks	perception-bias
SatCtr	url Sat CTR (with two different thresholds)	perception-bias
DSatCtr	url DSat CTR (with two different thresholds)	perception-bias
LastClkInSession	url last clicks in session / clicks	perception-bias
AveQueryClkCount	saverage number of clicks for query	query-bias
QueryClickRankDist	average (across query) sum of absolute difference between ranks for adjacent clicks	query-bias
QueryTimeBeforeFirstClk	average time before first click for the query	query-bias
QueryNoClkShows	query shows when there were no clicks	query-bias
AveQueryClkPos	query average click pos	query-bias
SatClkBeforeInQuery	the number of Sat clicks happened before the current url was clicked	query-bias
UrlClkMoreAveSessDwell	url clicks with more then average session dwell time	session-bias
UrlClkLongestSessDwell	url clicks with longest dwell time among session clicks	session-bias
AveSessionDuration	average session duration	session-bias
AveUrlRank	average url rank	sparsity
SmoothCtr	smoothed url CTR	sparsity
Query-Url, Url-Region, ...	back-off version of all above features	sparsity

Table 1: Example of features used and biases they target

used a couple of thresholds for dwell time to calculate Sat (clicks with dwell time more then some threshold) and DSat CTR (clicks with dwell time less then some threshold) of the document.

Query-Bias: User behavior varies for different queries. For example, 20 seconds of dwell time may indicate relevance for an easy query but irrelevance for a difficult one. To this end, we derive query features (e.g., mean/variation of click positions, average ctr, average time before first click, etc.) and normalize the URL features (e.g., dwell time) by query averages. For example, we calculated the number of times the document was clicked and dwell time was more then average for all clicks for this query.

Session-Bias: The session-bias encodes both behavioral variations from user and intent. Note that, intent is different from query as a query may carry multiple intents while an intent can be represented by different queries. To address this bias, we compute session features (e.g., session length) and normalize the URL features (e.g., click position in session) by the session averages. For example, we used all clicks in a session to calculate average/median dwell time and used this value as a threshold for the given document click dwell time. We also used the fact that a document was clicked and had the longest dwell time among all session clicks.

Sparsity: Behavior features are good indicators of document relevance for relatively frequent queries, but for infrequent queries, regions, URLs they tend to be sparse and therefore not very reliable. To address this issue we augment our feature set with back-off versions, namely in addition to query-region-url triple we also derive features for query-url, query, query-region, region-url and url. We also incorporate smoothed-CTR by adding pseudo counts to address sparsity.

Besides behavior features, the dataset gives us another kind of relevance evidence - original ranking. Even though we might not have enough click data to make a reasonable guess about document relevance, we still may use original ranking. So, we back-off to the original ranking (a proxy of clickthrough-orthogonal signals) when the clickthrough information is insufficient or not reliable (e.g., query is very difficult or infrequent). To represent original ranking, we use average rank position of URL. Together with the query features (e.g., frequency, CTR, averaged click position), we allow the machine learning algorithms to trade-off between the raw predictors and the various back-offs. Table 1 presents some of the features we used for our final submission.

5. LEARNING ALGORITHMS

In this section we describe two machine learning algorithms that we used to learn an optimal ranking function.

5.1 AUC-Rank

With this algorithm we aim to learn linear ranking function optimal in the sense of AUC metric on the training data. The *AUC-rank* is formulated as a regression problem, where we are trying to regress document relevance score given features derived from search engine log data. As was noted by previous work of [5] it is difficult to optimize AUC metric “as is” directly, since it involves calculation of indicator function 1 for pairwise comparison of candidate documents. We follow [5] in approximating AUC with sigmoid function and use this surrogate as our objective function.

$$AUC_{soft} = \frac{\sum_{i=0}^{p-1} \sum_{j=0}^{n-1} sigmoid_\beta(f(x_i, w) - f(y_j, w))}{pn} \tag{3}$$

$$sigmoid_\beta(x) = \frac{1}{1 + exp(-\beta x)} \tag{4}$$

where p is the number of relevant document, n is the number of not relevant documents for a given query, x_i is the i^{th} relevant document and y_j is the j^{th} not relevant document. For simplicity, we use linear function to calculate document score, i.e. $f(x, w) = \sum_{k=0}^{d} w_k x_k$, where $x \in X^d$ is feature vector corresponding to the URL in the training or test set and $w \in X^d$ is the weight vector. *AUC-rank* finds the weight vector w that maximizes AUC_{soft} on training set. The continuous and convex nature of our soft version of AUC allows us to optimize it efficiently using gradient descent algorithm. We use limited memory BFGS method [16] to choose the step size during the grading descent.

It is worth noting the fact that for queries where all documents have the same label - either relevant or not relevant the both AUC and AUC_{soft} are not defined. Therefore, we augment our objective function with regression like squared loss in order to make use of training data for queries with undefined AUC. Finally, to avoid overfitting we introduce $L2$-norm regularization on weight vector w. Later modification of the *AUC-Rank* referred as *AUC-Rank+Regression* in the result section.

5.2 Gradient Boosted Regression Trees

The second algorithm is GBRT (Gradient Boosted Regression Trees), which is a method for generating multiple weak learners (in our case, CART regression trees) and using them to get an aggregated predictor. It builds the model in a stage-wise manner like other boosting methods do, and it generalizes them by allowing optimization of an arbitrary differentiable loss function. The advantage of this non-linear regressor is the advanced expressiveness, which can help model the complex relationships among the features, but as a more complicated model it may not be applicable in certain large-scale scenarios. We use the pGBRT (Parallel Gradient Boosted Regression Trees) [18] implementation in our experiments.

6. METHODS COMPARED

We compare 21 models, including 2 baseline models representing the original ranking and the raw CTR (Clickthrough Rate), 3 models trained using all the features and the three different learning algorithms respectively, 8 models with all the back-off levels enabled but varying feature group combinations, and 8 models with all feature groups included but varying the combinations of different back-off levels.

Original Ranking Baseline: The first baseline (*OrigRank*) represents the original search engine ranking algorithm, which typically exploits various signals that are orthogonal to the behavioral features, such as link structure, document quality and query-document similarity. Specifically, for a given query-region-URL triple, the relevance score of an URL is computed as a weighted linear combination of the probabilities of the URLs ranked at different positions in response to the query-region pair, where higher weights are given to the higher ranks. The equation we use is given as below, where i represents the rank of an URL, $\#imprs_i$ represents the number of impressions at certain rank i, and $\#imprs$ represents the overall number of impressions.

$$score = \sum_{i=1}^{10} (10 - i) \frac{\#imprs_i}{\#imprs} \tag{5}$$

Raw CTR Baseline: The second baseline is the raw CTR (*RawCTR*), which is probably the most straightforward way of utilizing the clickthrough information. Given a query-region-URL triple, it is computed as $\#clks/\#imprs$, where $\#clks$ and $\#imprs$ represent the numbers that the URL was clicked and shown respectively in response to the query-region pair.

Full Model Runs: Three models are trained using all the feature groups and all the back-off levels enabled, namely, the logistic regression models (*AUC-Rank*), (*AUC-Rank+Regression*), and the gradient boosted regression trees full model (*GBRT_all*).

Single Feature Group Runs: Four models are trained using the gradient boosted regression trees with all the back-off levels enabled and with each single group of features, namely, the position-bias (*GBRT_position*), perception-bias (*GBRT_perception*), query-bias (*GBRT_query*), and session-bias (*GBRT_session*) groups.

Feature Ablation Runs: Four models are trained using the gradient boosted regression trees with all the back-off levels enabled and with each single group of features removed from the full model, namely, removing the position-bias group (*GBRT_no.position*), removing the perception-bias group (*GBRT_no.perception*), removing the query-bias group (*GBRT_no.query*), removing the session-bias group (*GBRT_no.session*).

Single Back-off Runs: Four models are trained using the gradient boosted regression trees with all the feature groups and with each single level of back-off enabled, namely, the query-region-url triple (*GBRT_query-region-url*), query-url pair (*GBRT_query-url*), region-url pair (*GBRT_region-url*), and the url features (*GBRT_url*)

Back-off Ablation Runs: Four models are trained using the gradient boosted regression trees with all the feature groups and with each single level of back-off removed, namely, removing the query-region-url triple (*GBRT_no.query-region-url*), removing the query-url pair (*GBRT_no.query-url*), removing the region-url pair features (*GBRT_no.region-url*), and the url features (*GBRT_no.url*)

7. RESULTS & DISCUSSIONS

In this section, we describe and discuss about our experimental results and findings. The results reported are average AUC's across 5-fold cross validation using the training set, as the test set for evaluating final submissions was not available at the time of this publication[1]. We start by presenting our main results by comparing our full models with the two baselines, and move on to comparing different feature groups and different back-off strategies. Finally, we discuss the effects of tuning the learning parameters of our non-linear full model and the importance of individual features.

The main results are summarized in Table 2. As we can see, all our three full models significantly outperform the two baselines, demonstrating the effectiveness of our proposed features in addressing the sparsity and various biases in the interaction data. The non-linear gradient boosted regression trees (GBRT) classifier outperforms the linear logistic regression AUC-Rank classifier, supporting our intuition that the GBRT is more powerful in modeling the complex interactions among different features. However, the linear AUC-Rank models are still useful in cases where datasets are large-scale and the more computationally expensive GBRT model might

[1] The absolute AUC numbers on the training set typically seem lower than those for the test set according to our experiments (e.g., one of our methods achieved AUC 0.626 for the training set and 0.636 for the test set in the leaderboard).

Method	*AUC*	*Improvement(%)*
GBRT_all	**0.6574**	**7.3**
AUC-Rank + Regression	0.6495	6.0
AUC-Rank	0.6337	3.4
RawCTR	0.6212	1.4
OrigRank	0.6126	n/a

Table 2: Average AUC in single run of 5-fold cross validation compared for baseline methods, AUC-rank and GBRT full models

Method	*AUC*	*Improvement(%)*
GBRT_all	0.6574	7.3
GBRT_session	0.6468	5.6
GBRT_perception	0.6430	5.0
GBRT_query	0.6412	4.7
GBRT_position	0.6307	3.0
OrigRank	0.6212	n/a

Table 3: Average AUC in 5-fold cross validation for models using features of single bias type

Method	*AUC*	*Improvement(%)*
GBRT_all	0.6574	7.3
GBRT_no.position	0.6555	7.0
GBRT_no.perception	0.6543	6.8
GBRT_no.session	0.6530	6.6
GBRT_no.query	0.6465	5.5
OrigRank	0.6126	n/a

Table 4: Impact of feature ablation on model performance, broken down by bias type

not be applicable. Interestingly, the *RawCTR* baseline achieves higher overall AUC than the *OrigRank* baseline, showing the important value of the user behavioral signals.

Single Feature Group Runs: The results are summarized in Table 3, with the different feature groups ranked in AUC-descending order. As we can see, all the single feature groups outperform the *RawCTR* baseline and under-perform the *GBRT_all* full model. Among the four different feature groups, the session-bias feature group (*GBRT_session*) performs the best, followed fairly closely by the perception-bias (*GBRT_perception*) feature group and query-bias (*GBRT_query*) feature group, and then followed by position-bias (*GBRT_position*) feature group with a bigger gap.

Feature Ablation Runs: The results are summarized in Table 4, with the different feature groups ranked in AUC-descending order. Again, all the feature combinations outperform the *RawCTR* baseline and under-perform the *GBRT_all* full model. Similar to the results in Table 3, the position-bias features seem to contribute the least, removing which only decreases the AUC from 0.6574 to 0.6555. Interestingly, removing the query-bias feature group decrease the performance most substantially, which is likely due to the weaker correlation of query feature group with the other groups of features. As a results, adding query feature group provides the most additional predictive power when the other groups are presented, even though by itself it is not the most predictive.

Method	*AUC*	*Improvement(%)*
GBRT_all	0.6574	7.3
GBRT_query-url	0.6435	5.0
GBRT_query-region-url	0.6367	3.9
GBRT_url	0.6360	3.8
GBRT_region-url	0.6313	3.1
OrigRank	0.6126	n/a

Table 5: Average AUC in 5-fold cross validation for models using back-off features

Single Back-off Group Runs: The results are summarized in Table 5. Intuitively, adding query and region information would increase the accuracy of estimating the url statistics, and ignoring these two factors would result in less accurate but potentially more robust statistics. As we can see, the most predictive individual back-off group is the query-url pair, followed by the query-region-url triple, url and region-url pair. The notion of relevance does not seem to vary much across different regions – ignoring region reduces the data sparsity and improves AUC in both the cases of query-region-url triple and the region-URL pair. In contrast, the notion of relevance seem to vary more substantially across different queries – keeping query information improves AUC significantly.

Back-off Group Ablation Runs: The results are summarized in Table 6. As we can see, all the back-off groups contribute to the full model. However, similar to what we have observed in Table 5, removing the region information does not influence the performance very significantly but removing the query information does.

Method	*AUC*	*Improvement(%)*
GBRT_all	0.6574	7.3
GBRT_no.region-url	0.6566	7.2
GBRT_no.url	0.6560	7.1
GBRT_no.query-region-url	0.6544	6.8
GBRT_no.query-url	0.6511	6.3
OrigRank	0.6126	n/a

Table 6: Impact of feature ablation on model performance, broken down by back-off level

Parameter Tuning for GBRT Algorithm: The GBRT algorithm has three parameters that one needs to specify prior to training - tree height (h), learning rate (λ), and number of boosting iterations (N). As the choice of particular values for these parameters is rather non trivial, we perform parameter sweeping and search for optimal parameter values in a grid manner. Based on our experiments, tree height of 3 achieves the optimal performance in most cases, thus, for simplicity, we fixed $h = 3$ and reduced our search space to learning rate and number of boosting iterations. For every value of N and λ we perform 5-fold cross validation and report average AUC across the 5 folds. Figure 1 summarizes the result of this experiment, where both z coordinate and the color correspond to the AUC in 5-fold cross validation. We vary λ with values $\{0.01, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.4, 0.5\}$ and N with values $\{50, 100, 200, 300, 400, 500, 1000, 2000\}$. We have found that with $\lambda = 0.1$ and $N = 200$ GBRT achieves the best AUC performance. The plot in Figure 1 confirms our intuition that smaller values of learning rate together with higher number of boosting iter-

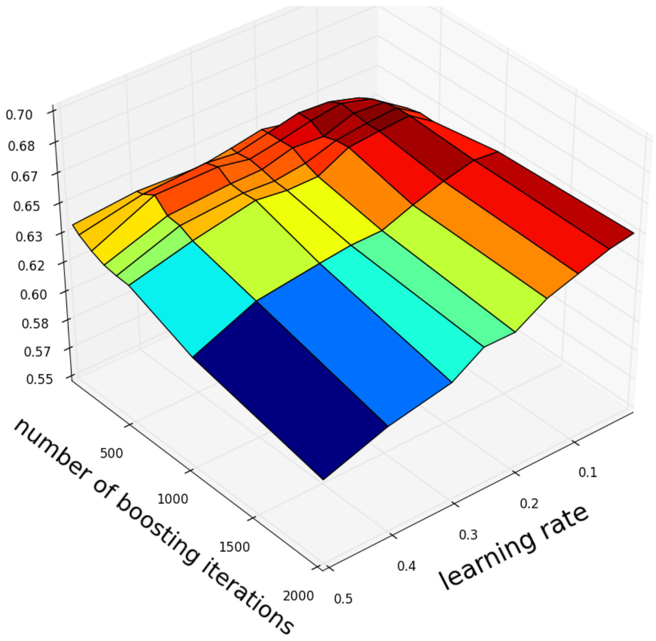

Figure 1: Average AUC in 5-fold cross validation plotted for different values of learning rate and boosting iterations

ations allow GBRT to make more accurate predictions and achieve better performance overall.

Feature Importance: We ranked all features by importance using χ^2 criterion (Weka implementation) and some representative features are given in Table 7. The features with the largest value of χ^2 criterion are those connected with url click dwell times (perception-bias). The feature ranked first by this criterion is the ratio of Sat clicks over overall number of clicks for query-url pair. This suggests that region isn't important for most of the queries and query-region-url level features are probably too sparse for some significant number of documents to make good predictions of document relevance. The feature ranked second is a back-off version of the previous feature. This suggests us that there were not so many popular urls in the dataset and aggregation of url features over all queries gives us a good estimate of relevance of this document to queries for which it was shown in top 10 results. Session level normalization features can be found in the top of the ranked features list. Features like clicks with more then average session dwell time and clicks dwell time normalized by the session duration seems to contribute to the document relevance. Features intended to deal with position-bias and query-bias were ranked in the middle of the list. The best features among those is url clicks when both documents around in the ranking were skipped / shows. The least values of the criterion belong to average click position kind of features and some query level features.

Feature Name	χ^2
Query-Url SatClk/TotalClk	2268.104
Url SatClk/TotalClk	2086.372
Query-Url DwellPerClk	2079.000
Query-Region-Url SatClk/TotalClk	2014.117
...	
Query-Url UrlClkMoreAveSessDwell	1981.104
Query-Url DwellPerSessDuration	1874.197
...	
Query-Url TotalClkCount	1844.585
Query-Url LastClkInSerp	1844.585
Url LastClkInSerp	1844.585
...	
Query-Region-Url DwellPerSessionDuration	1806.922
Query-Url LastClkInSession/Shows	1810.043
...	
Query-Url SkipClkAboveBelow	1706.362
...	
Query-Url SmoothCtr	1451.06
Query-Url # of SatClkBeforeInQuery	1402.967
...	
Query-Url AveUrlRank	710.3
...	
Query AveLastClkPos	0.0
Query-Region-Url AveClkPos	0.0

Table 7: Best individual features ranked by χ^2 statistics

8. CONCLUSIONS

With this paper we introduce a model capturing user behavior at different levels of granularity while eliminating four important types of bias from clickthrough data. Our experimental results show that the proposed behavioral features indeed provide additional and valuable information beyond original search result ranking and raw clickthrough rate, and outperform these two baselines in predicting editorial judgements of document relevance. Specifically, we found that post-click behavioral signals such as dwell time and the session-level information are among the most predictive signals of document relevance; we also found that the query-level signals are the most helpful among other features considered in this paper. In combination, our features allow for significant improvements of relevance prediction. Finally, we found that combination of URL statistics with respect to a query as well as its back-off versions such as overall URL statistics result in more robust and accurate predictive model.

9. REFERENCES

[1] M. Ageev, Q. Guo, D. Lagun, and E. Agichtein. Find it if you can: A game for modeling different types of web search success using interaction data. In *Proceeding of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval(SIGIR 2011)*, 2011.

[2] E. Agichtein, E. Brill, and S. Dumais. Improving web search ranking by incorporating user behavior information. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '06, pages 19–26, New York, NY, USA, 2006. ACM.

[3] A. Aula, R. M. Khan, and Z. Guan. How does search behavior change as search becomes more difficult? In *Proceedings of the 28th international conference on Human factors in computing systems*, CHI '10, pages 35–44, New York, NY, USA, 2010. ACM.

[4] A. Broder. A taxonomy of web search. *SIGIR Forum*, 36:3–10, September 2002.

[5] T. Calders and S. Jaroszewicz. Efficient auc optimization for classification. *Knowledge Discovery in Databases: PKDD 2007*, pages 42–53, 2007.

[6] N. Craswell and M. Szummer. Random walks on the click graph. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '07, pages 239–246, New York, NY, USA, 2007. ACM.

[7] N. Craswell, O. Zoeter, M. Taylor, and B. Ramsey. An experimental comparison of click position-bias models. In *Proceedings of the international conference on Web search and web data mining*, WSDM '08, pages 87–94, New York, NY, USA, 2008. ACM.

[8] G. Dupret and C. Liao. A model to estimate intrinsic document relevance from the clickthrough logs of a web search engine. In *Proceedings of the third ACM international conference on Web search and data mining*, WSDM '10, pages 181–190, New York, NY, USA, 2010. ACM.

[9] G. E. Dupret and B. Piwowarski. A user browsing model to predict search engine click data from past observations. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '08, pages 331–338, New York, NY, USA, 2008. ACM.

[10] J. Gao, W. Yuan, X. Li, K. Deng, and J.-Y. Nie. Smoothing clickthrough data for web search ranking. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '09, pages 355–362, New York, NY, USA, 2009. ACM.

[11] F. Guo, C. Liu, A. Kannan, T. Minka, M. Taylor, Y.-M. Wang, and C. Faloutsos. Click chain model in web search. In *Proceedings of the 18th international conference on World wide web*, WWW '09, pages 11–20, New York, NY, USA, 2009. ACM.

[12] F. Guo, C. Liu, and Y. M. Wang. Efficient multiple-click models in web search. In *Proceedings of the Second ACM International Conference on Web Search and Data Mining*, WSDM '09, pages 124–131, New York, NY, USA, 2009. ACM.

[13] Q. Guo, R. W. White, S. T. Dumais, J. Wang, and B. Anderson. Predicting query performance using query, result, and user interaction features. In *RIAO '10: Adaptivity, Personalization and Fusion of Heterogeneous Information*, pages 198–201, Paris, France, France, 2010. LE CENTRE DE HAUTES ETUDES INTERNATIONALES D'INFORMATIQUE DOCUMENTAIRE.

[14] B. Hu, Y. Zhang, W. Chen, G. Wang, and Q. Yang. Characterizing search intent diversity into click models. In *Proceedings of the 20th international conference on World wide web*, WWW '11, pages 17–26, New York, NY, USA, 2011. ACM.

[15] C. Ling, J. Huang, and H. Zhang. Auc: a statistically consistent and more discriminating measure than accuracy. In *International Joint Conference on Artificial Intelligence*, volume 18, pages 519–526. LAWRENCE ERLBAUM ASSOCIATES LTD, 2003.

[16] J. Nocedal. Updating quasi-newton matrices with limited storage. *Mathematics of computation*, 35(151):773–782, 1980.

[17] F. Radlinski and T. Joachims. Query chains: Learning to rank from implicit feedback. In *ACM SIGKDD International Conference On Knowledge Discovery and Data Mining (KDD)*, pages 239–248, 2005.

[18] S. Tyree, K. Q. Weinberger, K. Agrawal, and J. Paykin. Parallel boosted regression trees for web search ranking. In *Proceedings of the 20th international conference on World wide web*, WWW '11, pages 387–396, New York, NY, USA, 2011. ACM.

[19] F. Wilcoxon. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6):80–83, 1945.

[20] Y. Zhang, W. Chen, D. Wang, and Q. Yang. User-click modeling for understanding and predicting search-behavior. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '11, pages 1388–1396, New York, NY, USA, 2011. ACM.

[21] F. Zhong, D. Wang, G. Wang, W. Chen, Y. Zhang, Z. Chen, and H. Wang. Incorporating post-click behaviors into a click model. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '10, pages 355–362, New York, NY, USA, 2010. ACM.