User Behavior Driven Ranking without Editorial Judgments

Taesup Moon, Georges Dupret
Yahoo! Labs, 701 First Ave, Sunnyvale, CA 94089
{taesup, gdupret}@yahoo-inc.com

Shihao Ji, Ciya Liao [*]
Microsoft
One Microsoft Way, Redmond, WA 98052
{shihaoji, cliao@microsoft.com

Zhaohui Zheng
Yahoo! Labs
701 First Ave, Sunnyvale, CA 94089
zhaohui@yahoo-inc.com

ABSTRACT

We explore the potential of using users click-through logs where no editorial judgment is available to improve the ranking function of a vertical search engine. We base our analysis on the *Cumulated Relevance Model*, a user behavior model recently proposed as a way to extract relevance signal from click-through logs. We propose a novel way of directly learning the ranking function, effectively by-passing the need to have explicit editorial relevance label for each query-document pair. This approach potentially adjusts more closely the ranking function to a variety of user behaviors both at the individual and at the aggregate levels. We investigate two ways of using behavioral model; First, we consider the parametric approach where we learn the estimates of document relevance and use them as targets for the machine learned ranking schemes. In the second, functional approach, we learn a function that maximizes the behavioral model likelihood, effectively by-passing the need to estimate a substitute for document labels. Experiments using user session data collected from a commercial vertical search engine demonstrate the potential of our approach. While in terms of `DCG` the editorial model out-performs the behavioral one, online experiments show that the behavioral model is on par –if not superior– to the editorial model. To our knowledge, this is the first report in the Literature of a competitive behavioral model in a commercial setting.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval; I.2.6 [**Articial Intelligence**]: Learning

General Terms

Algorithms, Experimentation

[*]This work was done while the authors were with Yahoo! Labs.

Keywords

Click-Through Data, User Behavior, Search Engines, Probabilistic Model

1. INTRODUCTION

It is quite natural to assume that users' interactions with the result pages of a search engine convey information about the degree of relevance of the web pages. This idea has inspired many researchers to use click-through logs to improve user search experience. The main approach was to show how useful an appropriate interpretation of user interactions –clicks and query reformulations, in particular– can help in constructing more performant relevance functions. On the other hand, little work has been published on the problem of learning a ranking solely based on user interactions, entirely by-passing editorial judgment of editors. More precisely, we formulate the problem as follows; Start with a reasonable Information Retrieval ranking function like BM25 or a good language model based approach, collect clicks and use this information to bootstrap a full fledge, state-of-the-art function able to compete with the best learning to rank algorithms based on tens of thousands of query-document judgment pairs.

State of the art search engines use various algorithms that extend the supervised machine learning framework to ranking. These have become popular approaches because they have both reasonable complexities and good scalability for very large size of Web data. One caveat of learning to rank methods, however, is that they require expensive manual labeling of documents to generate training data. The problem is multifold: one is that the labels needs to be updated and expanded regularly in order to maintain a large size of training data and cope with the time-varying, dynamic nature of the Web. The other is that those labor intensive relevance judgments may contain errors and be different from what the real users on the Web require. The problem is made worse, as the internet expands, by the need for ever more specialized search engines –called *verticals* for which it is impractical, time consuming, expensive and even infeasible to gather enough editorial judgment. For example, in this work we carry the numerical experiments on the data of a local search engine where it is hard for an editor to judge if such business is a good recommendation for a user living in a remote neighborhood he is unfamiliar with.

Contribution. In this paper, we explore two related ways of exploiting the record of user interactions with the search engine. Namely, the methods we propose attempt to learn a ranking function directly from the user behavior data, instead of editorial judgment data. The basic assumption is that the user behavior is influenced by the overall relevance of the search result sets, and such behaviors can be captured by probabilistic models, treating the relevance of the search result sets as hidden variables. Once a probabilistic model has been decided upon, we have two possibilities:

- We can use the click-through data to estimate the document relevance and use these as labels for a learning to rank algorithm,
- We can relate those relevance-based hidden variables with functions defined on a set of features and learn the relevance functions by finding the function that maximizes the likelihood of the probabilistic model.

The second approach is more novel and we describe the general functional learning framework it belongs to in Section 3. The rest of the paper is organized as follows. In Section 2, we review some prior work. Then, in Section 4, we summarize a particular user behavior model, the *Cumulated Relevance model*, which was recently introduced in [3] and that serves as a basis for further experiments. Section 4.1 will explain the exact procedure for using the *Cumulated Relevance model* and functional gradient boosting method to obtain user behavior driven ranking functions. Section 5 reports experimental results, and Section 6 concludes with a discussion and future research directions.

2. RELATED WORK

Over the past decade, a large number of learning to rank algorithms, which require human editorial judgment labels, have been proposed, see [7] and references therein. They mainly focus on devising sensible loss functions as surrogates to the true ranking loss so that minimizing such loss functions would lead to a good ranking performance. The user behavior modeling has been primarily centered around modeling click-through logs of search engines. We can divide in essentially four categories according to how click information has been used in the literature: gaining insight into typical user behaviors [6], estimating the relevance feature or target values for documents from the user clicks [9, 2], deriving a metric based on clicks to compare ranking functions [1], and directly re-ranking the top documents retrieved from search engine [5]. By no means, these lists are exhaustive.

3. USER BEHAVIOR DRIVEN RANKING

In this section, we describe the general framework for learning a ranking function both in parameter and functional space. We use user sessions as the basic data structure to record user behavior as well as the content of the result set. To this end, we use S to denote a user session, and it consists of three components: a sequence of queries $Q = \{q_i\}$, a set of documents $D = \{d_{ij}\}$ presented in response to the queries, and user interactions with those documents in the form of clicks $C = \{c_{ij}\}$, where c_{ij} is 1 if the j-th document of query i was clicked and 0 otherwise. Namely, we can express the session S as $S \triangleq \{Q, D, C\}$. Once a user session S is given, we can associate the collection of events E that are generated from the session to describe the content of the result sets and user behaviors. In addition to associating the observable events with the session S, we also assume a set of hidden variables H that capture aspects that are not directly observable. In this work, our set of hidden variables only includes document relevance, but the framework we propose is by no means limited to this choice.

The *user behavior models* (UBM) can then be derived from the session data S by imposing a joint probabilistic model on the event $\mathcal{E}$ and the hidden variables H, parameterized by the parameter set U, i.e., $P(\mathcal{E}, H|U)$. In existing work, the main usage of such models was to take advantage of abundant user session data $S^N = \{S_1, \dots, S_N\}$ and estimate the parameters of the model by maximizing the likelihood of the observations $\mathcal{E}^N = \{\mathcal{E}_1, \dots, \mathcal{E}_N\}$. This takes the form of finding the set of parameters U that maximize the following expression with respect to the parameters in U: $\prod_{n=1}^{N} \sum_{H_n} p(\mathcal{E}_n, H_n|U)$, and such estimated values of U were used as relevance features for ranking functions.

We can go one step further in this line of reasoning and instead of learning the values of the parameters in U, we consider these as functions of a set of relevant features. To fix ideas, we postulate for a query and document pair (q_i, d_{ij}) the existence of a corresponding parameter $u_{ij} \in U$ that represents the degree of relevance of document d_{ij} to query q_i. We further assume that this relevance can be estimated as a function of features that characterizes the query document pair: $u_{ij} = f(x_{ij})$, where $x_{ij} \in \mathbb{R}^d$ represents a d-dimensional feature vector for (q_i, d_{ij}), and f is a function that maps $\mathbb{R}^d$ into $\mathbb{R}$. Then, we can directly learn the relevance function by finding f that maximizes $\prod_{n=1}^{N} \sum_{H_n} p\big(\mathcal{E}_n, H_n|\{f(x_{ij}^{(m)})\}\big)$, and sorting documents according to the values of f for a given query. We try to show the effectiveness of such obtained ranking function via experiments.

In the next sections, we illustrate the framework both in parametric and functional space on the *Cumulated Relevance Model*, a particularly simple model of the influence of document relevance on user behavior.

4. CUMULATED RELEVANCE MODEL

Here, we briefly summarize the original cumulated relevance model. For simplicity, we only consider the case where each session consists of a single query q. Then, without loss of generality, we can express $D = \{d_i\}$ and $C = \{c_i\}$, and the size of both sets n. We then denote $\mathcal{I}(C) \triangleq \{i : c_i \in C,\ c_i = 1\}$ as the set of clicked document indices in C and, for all $i \in \mathcal{I}(C)$, denote $\mathcal{I}(C)^i \triangleq \{j : c_j \in C, c_j = 1, j \leq i\} \subseteq \mathcal{I}(C)$ as the set of clicked document indices up to i-th document. Given these notations, for given N sessions, the cumulated relevance model tries to predict the session success and estimates the document utilities by minimizing $L(u_0, u_1, \dots, u_n) = \sum_{m=1}^{N} \sum_{i \in \mathcal{I}(C_m)} \log\big(1 + \exp\big(\ell_i\big(u_0 + \sum_{j \in \mathcal{I}(C_m)^i} u_j\big)\big)\big)$ with $\ell_i = 1 - 2s_i$. For more details, refer to [3]. They also partially solve the sparsity problem by adding a Gaussian priors with mean μ_u and variance σ_u^2 to the utilities; they introduce regularization terms and modifies the original loss function as

$$\underset{u_1,\dots,u_n}{\text{minimize}} \left(L(u_0, u_1, \dots, u_n) + \sum_{i=1}^{n} \frac{(u_i - \mu_u)^2}{2\sigma_u^2} \right). \qquad (1)$$

which becomes a maximum *a posteriori* estimation.

4.1 Functional learning

Once the loss function is defined as in the previous section, we can use the step described in Section 3 to express it in a functional form. That is, by associating the utility variable as $u_i = f(x_i)$, where x_i is the feature vector for i-th example, we convert the loss function into $L(u_0, f)$, which equals to

$$\sum_{m=1}^{N} \sum_{i \in \mathcal{I}(C_m)} \log\Big(1 + \exp\Big(\ell_i\big(u_0 + \sum_{j \in \mathcal{I}(C_m)^i} f(x_j)\big)\Big)\Big), \quad (2)$$

and try to minimize it in a functional space to obtain the ranking function. We can also add the regularization term as in (1). Since the loss function (2) takes into account the whole document list to compute the loss, it can be seen as a listwise loss function. The optimization of the loss functions can be carried out by using functional gradient descent method [4] with using regression trees as weak learners, which results in the ranking function.

5. EXPERIMENTS ON LOCAL SEARCH

We now present the experimental results on a commercial local search engine. , i.e. a specialized search engine dedicated to find business of different kinds like restaurant, hardware, or flower shops, etc. in the neighborhood of the user. First, we describe how we collected data, then describe the performance metric we use. Finally, we present some results to demonstrate the effectiveness of the user behavior driven ranking functions.

5.1 Data and performance metric

Click Session Data: We collected 1.8 million sessions with at least on click from the local search engine, which resulted in 3 million query-document pairs. Each query-document pair (q, d) is represented with a feature vector x, of which dimension is more than 500.

Editorial Judgment Data: To evaluate the performance of ranking functions derived from the different approaches, we also gathered a set of editorial judgment data. We randomly sampled 13,916 queries from the query log, and for each query up to the top 15 documents are retrieved. This results in 170,431 query-document pairs. We then requested human editors to label each query-document pair to be {Perfect, Excellent, Good, Fair, Bad} according to the degree of relevance of the document with respect to the query. Since we need to train the editorial models, the data is further divided randomly into training set and test set, with 11,583 queries (or 141,238 query-document pairs) in training set and 2,333 queries (or 29,193 query-document pairs) in the test set.

We adopt Discounted Cumulative Gain (DCG) as our editorial metric because of its popularity and its general acceptance. The DCG at position 5 (DCG@5) of a ranked list of documents with respect to a query q is defined as a weighted average of the scores of the documents in the ranking: $\text{DCG@5} := \sum_{r=1}^{5} \frac{2^{g_r}-1}{\log(1+r)}$, where $g_r \in \{0, 1, 2, 3, 4\}$ is the relevance score of the r-th document in the ranked result set. $g_r = 4$ corresponds to a "perfect" label, and $g_r = 0$ corresponds to a "bad" label. Note that this metric would be naturally biased toward the editorial judgment based ranking function. However, we still use this metric for our offline experiments as a sanity check.

5.2 Results

We now move on to demonstrate the performances of the user behavior driven learning. We compare various behavioral and editorial model implementations.

5.2.1 *Offline experimental result*

We compared the following six different ranking functions:

Scheme 1: Baseline. This is a hand-tuned ranking function deployed in the local search engine for several years. The click session data used in the experiments was collected from the users in response to the rankings generated from this ranking function;

Scheme 2: State-of-the-art learning to rank algorithms that are based on editorial judgment data, such as GBDT [4] and GBRank [9]. They are collectively refered to as Editorial Machine Learned Ranking or "Editorial MLR";

Scheme 3: 1-step implementation of user behavior driven learning, i.e., directly optimizing (2) via functional gradient boosting trees;

Scheme 4: 2-step implementation of user behavior driven learning, i.e., firstly, estimate the document utilities by optimizing (1) in parametric space, and then treat the derived utilities as targets and apply GBDT or GBRank to train a ranking function;

Scheme 5: Pair-wise learning based on "Skip Above" and "Skip Next" pairs [8]. We extract pairwise preference between documents from the click-through data. These pairs then serve as labels for the GBRank algorithm;

Figure 1 provides an overall performance comparison in terms of DCG. The different curves denote the best performances we achieve with different parameter sets in the gradient boosting trees method and in the model. All algorithms are limited to a maximum of 800 trees, which is enough for all of them to converge. The editorial models training and test sets are the product of randomly sampling the queries with editorial judgments. Training sample size is varied and the effect on performance on the test set is shown in Figure 1 where the x-axis reports the proportion of the total set used for training.

It is demonstrated in Figure 1 that

- As the size of training set increases, the editorial model shows an increase in DCG@5 values on the test set. Compared to the baseline, there is about 1.8% DCG@5 at one end, and 7.9% DCG5@5 gain at the other end.
- Scheme 3 yields a model with 1.9% DCG@5 gain over the baseline, almost on par with the "Editorial MLR" with 1% of training data.
- Surprisingly, the 2-step implementation of user behavior learning, Scheme 4, with GBDT is worse by 1% than the baseline in DCG terms.
- On the other hand, the 2-step implementation, Scheme 4, with GBRank has 4.9% DCG@5 gain over the baseline. This is about 3% DCG@5 gap to the best "Editorial MLR" trained on the full data set.
- Skip Above & Skip Next is slightly below production. It is probably possible to optimize further this method, but it is not too promising when compared with the other methods. It serves essentially as an extra sanity check.

Overall, both the 1 and 2-step methods improve over the baseline, which shows their viability. The 1-step implementation and 2-step GBDT optimize similar log-likelihood functions (2) and (1), respectively, we would expect similar performance. An important difference between the two is that the utilities of the documents are learned independently for each query in the 2-step method while they are learned together in the 1-step one. This entails that they are constrained to be on the same scale in the 1-step method,

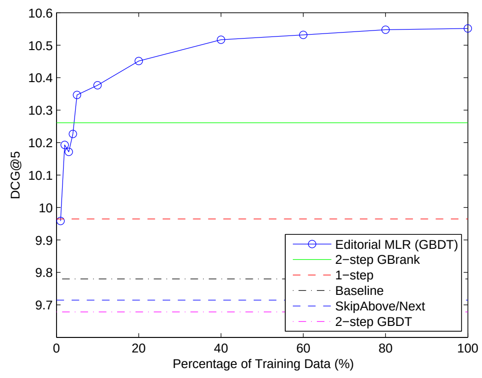

Figure 1: Performance comparison among different algorithms.

but not in parametric space. When learning the parametric utilities, GBDT, being a point-wise method, enforce the scores to share a common scale, while GBRank, being a pair-wise method, is sensitive only to the score differences. This means that GBRank is more flexible and this might explain why it performs better. This also suggests that we should revise the *Cumulated Relevance model* to work on difference in utility rather than absolute values.

5.2.2 *Online experimental result*

Ranking functions can be compared in terms of DCG, but they can also be compared on how users interact with them. To gain more insight, we select the editorial model and the behavioral model with highest DCG (2-step GBRank) and diverted part of the live traffic from the production function towards the two functions we want to test. Figure 2 reports the click-through rate (CTR) of both models. We observe that the behavioral model experiences a slightly larger, statistically significant, CTR at all positions. The average click rank of the behavioral model is 3.85 against 3.94 for the editorial model. The average CTR at positions 1 and 2 are larger by 3.4% and 1.8% respectively. Although simply comparing CTR is a tricky comparison, we believe improving clicks on all positions clearly is a positive signal for ranking quality. As far as we know, this is the first record in the Literature of a purely user behavior model showing performance on par or better than an editorial model in comercial settings. This also demonstrates the limitations of DCG.

6. CONCLUSION AND FUTURE WORK

We approached the problem of using the click-through logs of user interactions with a search engine to construct a new ranking function that adapts to users. The goal is ambitious as we want this new function to show comparable performance to a state-of-the-art ranking algorithm trained on tens of thousands query-document of editorial judgments. We setup a general framework and used the cumulated relevance model as a concrete example to carry out experiments. Although the DCG of the editorial based model was higher than the user behavior model in the offline experiment, we observed that in the online experiments, users seem to prefer it. To our knowledge, this is the first report in the Literature of a ranking function trained exclusively on click data that outperforms or is at least on par with the state-of-the-art, heavily trained editor based ranking function. As for future work, we plan to relax some assumptions of the cumulated relevance models to further improve the performance of the user behavior driven ranking models.

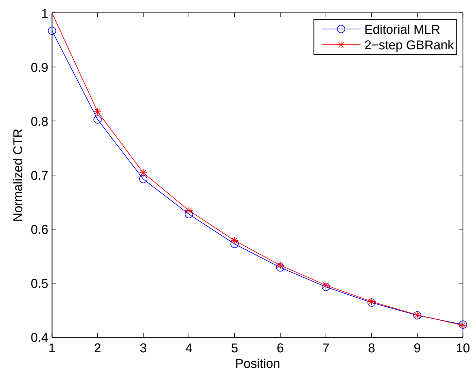

Figure 2: Click-through rate for Editorial and Behavioral models in online bucket tests. Values are normalized by the largest CTR.

7. REFERENCES

[1] B. Carterette and R. Jones. Evaluating search engines by modeling the relationship between relevance and clicks. *Advances in Neural Information Processing Systems*, 20:217–224, 2008.

[2] O. Chapelle and Y. Zhang. A dynamic bayesian network click model for web search ranking. In *WWW '09: Proceedings of the 18th international conference on World wide web*, pages 1–10, New York, NY, USA, 2009. ACM.

[3] G. Dupret and C. Liao. Cumulated relevance: A model to estimate document relevance from the clickthrough logs of a web search engine. In *Proceedings of the third International ACM Conference on Web Search and Data Mining (WSDM)*, 2010.

[4] J. Friedman. Greedy function approximation: a gradient boosting machine. Technical report, Stanford University, 1999.

[5] S. Ji, K. Zhao, C. Liao, Z. Zheng, G. Xue, O. Chapelle, G. Sun, and H. Zha. Global ranking by exploiting user clicks. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 35–42, 2009.

[6] T. Joachims, L. Granka, B. Pan, H. Hembrooke, and G. Gay. Accurately interpreting clickthrough data as implicit feedback. In *Proceedings of ACM SIGIR 2005*, pages 154–161, New York, NY, USA, 2005. ACM Press.

[7] T. Y. Liu. *Learning to Rank for Information Retrieval.* Now Publishers, 2009.

[8] F. Radlinski and T. Joachims. Evaluating the robustness of learning from implicit feedback. In *ICML Workshop on Learning In Web Search*, 2005.

[9] Z. Zheng, H. Zha, K. Chen, and G. Sun. A regression framework for learning ranking functions using relative relevance judgments. In *Proccedings of Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2007.