Global Ranking by Exploiting User Clicks

Shihao Ji
Yahoo! Labs
701 First Avenue
Sunnyvale, CA 94089
shihao@yahoo-inc.com

Ke Zhou
Dept. of Computer Science and Engineering
Shanghai Jiao-Tong University
zhouke@apex.sjtu.edu.cn

Ciya Liao, Zhaohui Zheng
Yahoo! Labs
701 First Avenue
Sunnyvale, CA 94089
ciyaliao, zhaohui@yahoo-inc.com

Gui-Rong Xue
Dept. of Computer Science and Engineering
Shanghai Jiao-Tong University
grxue@apex.sjtu.edu.cn

O. Chapelle, Gordon Sun
Yahoo! Labs
701 First Avenue
Sunnyvale, CA 94089
chap, gzsun@yahoo-inc.com

Hongyuan Zha
College of Computing
Georgia Tech.
Atlanta, GA 30032
zha@cc.gatech.edu

ABSTRACT

It is now widely recognized that user interactions with search results can provide substantial relevance information on the documents displayed in the search results. In this paper, we focus on extracting relevance information from one source of user interactions, i.e., user click data, which records the sequence of documents being clicked and not clicked in the result set during a user search session. We formulate the problem as a global ranking problem, emphasizing the importance of the sequential nature of user clicks, with the goal to predict the relevance labels of *all* the documents in a search session. This is distinct from conventional learning to rank methods that usually design a ranking model defined on a single document; in contrast, in our model the relational information among the documents as manifested by an aggregation of user clicks is exploited to rank all the documents *jointly*. In particular, we adapt several sequential supervised learning algorithms, including the conditional random field (CRF), the sliding window method and the recurrent sliding window method, to the global ranking problem. Experiments on the click data collected from a commercial search engine demonstrate that our methods can outperform the baseline models for search results re-ranking.

Categories and Subject Descriptors

H.3.3 [**Information Systems**]: Information Search and Retrieval—*Relevance Feedback*; H.4.m [**Information Systems**]: Miscellaneous—*Machine learning*

General Terms

Algorithms, Experimentation, Human Factors

Keywords

Learning to rank, implicit relevance feedback, user clicks, sequential supervised learning, conditional random field, experimental evaluation

1. INTRODUCTION

To a large extent, the quality of a search engine is determined by the ranking functions the search engine deploys. The key is to design a set of features or descriptors to represent a query-document pair that are good indicators of the degree of relevance of a document with respect to a query. There are many data sources that are explored in building the ranking functions. In the early days, information retrieval systems have seen heavy reliance on exploring textual data. For example, the feature-oriented probabilistic indexing methods use textual features such as the number of query terms, length of the document text, term frequencies for the terms in the query to represent a query-document pair [8]; the vector space models use the raw term and document statistics to compute the similarity between a document and a query [18]. With the advance of the World Wide Web, a new generation of systems utilize the hyperlink structures of Web documents, among them are those based on PageRanks and anchor texts which substantially contributed to the popularity of the Google search engine [1]. The race to discover the next gold mine of data sources and powerful features extracted from them for search engines is still on-going, and much recent research has focused on exploring user interactions with search results, i.e., user click data, to extract relevance information [9, 10, 17, 22, 3, 4, 6].

Parallel to the exploration of new data sources, ranking function design has also experienced a rapid development in recent years. In the framework of learning to rank, several state-of-the-art machine learning based ranking algorithms have been proposed, including RankSVM [9], RankNet [2] and GBrank [23]. Although these ranking algorithms are quite different in terms of ranking models and optimization techniques, all of them can be regarded as “local ranking”, in the sense that the ranking model is defined on a single document. Specifically, in “local ranking” the ranking score of a document is given largely based on the feature vector of the *current* document without considering the possible relations to other documents to be ranked. For many applications, this is only a loose approximation as relational information among documents always exists, e.g., in some cases two similar documents are preferred having similar relevance scores, and in other cases a parent document should be potentially ranked higher than its child documents. Thus, more advanced ranking algorithms should utilize all the information (both *local* and *global*) together, and define the ranking model as a function on all the documents to be ranked [14]. This may be even more true when one wants to extract relevance information from user click data since users' click decisions among different documents displayed in a search session tend to rely not only on the relevance judgement of a single document, but also on the *relative* relevance comparison among the documents displayed; and user click sequences can be a substantial relevance indicator of the relevance labels of the documents with regard to the query.

Towards developing a reliable click modeling method, this paper focuses on extracting relevance information from user click data via global ranking, which is explored here to utilize the relational information among the documents as manifested by user clicks. In particular, we introduce a global ranking framework of modeling user click sequences, and adapt several sequential supervised methods, such as the conditional random fields (CRF) [12], the sliding window method and the recurrent sliding window method [5], to this click modeling problem.

The rest of the paper is organized as follows. In Section 2, we provide an overview of related work in global ranking and relevance extraction from user click data. In Section 3, we give a formal definition of global ranking, with a comparison to the conventional learning to rank methods. In Section 4, we illustrate why sequential correlations in user click data are important to infer relevance information, and how the click features are extracted to summarize these information for the global ranking problem. In Section 5, we explore several sequential supervised learning algorithms to the global ranking problem; we emphasize how to adapt these methods in respect to the ranking nature of Web search. In Section 6, we carry out an extensive experimental study using the click data from a commercial search engine, with the comparison among different sequential supervised learning methods and several unsupervised methods proposed in the literature.

2. RELATED WORK

Global ranking is an explored idea in many ranking related research. To our knowledge, the first formal definition of global ranking was given by Qin et al. in [14], where the authors also proposed two applications of it in Pseudo Relevance Feedback and Topic Distillation. In contrast to our work, the relational information between the documents that is exploited in [14] is either document similarity or parent-child relations, while in this paper we focus on exploiting the relational information as manifested by an aggregation of user clicks. In addition, the modified CRF algorithm in [14] does not tackle directly a ranking problem, in which the absolute relevance grades are not important, but only the score ranks matter.

There are also a great deal of work exploring click data to extract relevance information. For example, Craswell et al. [4] and Dupret et al. [6] investigate several generative probabilistic models for user clicks, and aim to simulate human click behaviors in search results. However, the click features used in their work are relative simple, and both methods are in the framework of unsupervised learning (i.e., no human judgements are required in these methods for information extraction). It is well-known that user clicks are inherently noisy; by exploring supervised learning in click data modeling as in this paper, we expect our click model can reliably extract relevance information by calibrating with human relevance judgments. Probably, the closest work to our approach is that of Carterette and Jones [3], in which the authors use raw click frequencies to predict the absolute relevance labels. However, in their method the labels are predicted independently. As discussed in the introduction, this may not fully exploit the user click information as that can be utilized by global ranking.

3. GLOBAL RANKING PROBLEM

Global ranking was first formally introduced by Qin et al. in [14]. Independently developed by us, the click modeling method we proposed in this paper is essentially in the same framework of global ranking. We therefore use this terminology for better clarification of the basic ideas.

Let $\mathbf{x}^{(q)} = \{x_1^{(q)}, x_2^{(q)}, \ldots, x_n^{(q)}\}$ represent the documents retrieved with a query q, and $\mathbf{y}^{(q)} = \{y_1^{(q)}, y_2^{(q)}, \ldots, y_n^{(q)}\}$ represent the relevance labels assigned to the documents. Here n is the number of documents retrieved with q. Without loss of generality, we assume in this paper that n is fixed and invariant with respect to different queries. In the framework of supervised learning, $\mathbf{y}^{(q)}$ is assigned by human judges in the training phase, and is determined by a ranking model in the testing phrase.

If a ranking model is defined on a single document, i.e., in the form of

$$y_i^{(q)} = f(x_i^{(q)}), \quad \forall i = 1, \ldots, n, \tag{1}$$

it is referred as "local ranking". Otherwise, if a ranking model takes all the documents as its inputs and exploits both local and global information among the documents, i.e., in the form of

$$\mathbf{y}^{(q)} = F(\mathbf{x}^{(q)}), \tag{2}$$

it is referred as "global ranking".

Apparently, most of recent learning to rank algorithms, such as RankSVM [9], RankNet [2] and GBrank [23], are in the category of "local ranking". To our knowledge, the first global ranking algorithm could be the one proposed in [14], where the CRF [12] is modified to adapt to the ranking problem. As we will discuss in Section 5, this modified CRF algorithm does not tackle the ranking problem directly but more in the sense of regression, and we will address this issue in details in Section 5. In the next section, we will discuss why the sequential correlations in user clicks are important to infer relevance information, and how we can exploit them in global ranking.

4. SEQUENTIAL CORRELATIONS AS MANIFESTED IN USER CLICKS

We first briefly describe the user click data that are used in our study. The data are collected from a commercial search

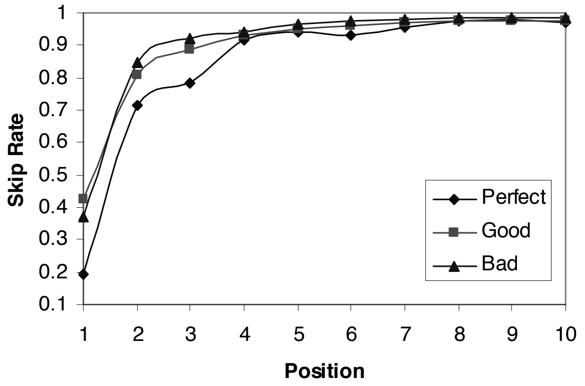

Figure 1: The skip rate of a document (relevant or irrelevant) as a function of position in the result set.

engine for a certain period of time. We first preprocess the raw click logs to extract all the user click sessions, where each session contains the query submitted to the search engine, the documents displayed in the result set, and the click information indicating whether a document is clicked or not, and the click time stamps. We only examine the top ten documents in each user click session, this usually corresponds to the documents displayed in the first page of the result set. Furthermore, we aggregate all the user sessions that have the same query, from which we keep the user sessions that have the most frequent top ten documents[1], and eliminate all the other user sessions. We call the tuple <query, 10-document list, and an aggregation of user clicks> an aggregated session. In this way we can ensure that there is an *unique* aggregated session for each query in the dataset. For the purpose of experimental study, each query-document pair is assigned a label from an *ordinal* set

`{perfect, excellent, good, fair, bad}`

to indicate the degree of relevance of the document with respect to the query in question. This allows us to calculate some click statistics and analyze user click behaviors.

Figure 1 shows the average number of sessions for a query, in which a document at a certain position is skipped (not clicked), out of all the sessions for the query (a.k.a. skip rate). We consider the skip rates for three relevance grades: `perfect, good, bad`. We observe that the skip rates are substantially higher for documents at the bottom of the result set regardless of the relevance grades of the documents. Documents with `perfect` grade generate more clicks at the top positions, but documents with `bad` grade also garner substantial clicks on par with those with `good` grade. This demonstrates that users tend to click the top documents even though the relevance grades of the documents are low and the raw click frequencies alone will not be a reliable indicator of relevance.

Further investigation leads us to focus on the sequential nature of user clicks. Table 1 provides a typical example extracted from the dataset with regard to the query: `pregnant man`. The first line of the table includes the query issued to the search engine, and each following line lists the sequence

[1]In response to a query, search engines sometimes may return the top ten documents in varying orders, or some new documents may appear in the top ten list due to search infrastructure changes and/or ranking feature updating.

Table 1: Sample user click sessions for the query "pregnant man".

[pregnant man]
2 3 5
1 2 3
2 3
1 3
1 2 3
2 3 6 7 3

of clicks a user performed in his/her query session, and the numbers in the table denote the positions of the document clicked in the result set. We examine the second and the third documents, which are labeled as `good` and `excellent`, respectively. The second document:

`http://abcnews.go.com/Primetime/story?id=2346476&page=1`

is an ABC news article in August 2006 about a medical mystery: A man in India with twin living inside him. The third document

`http://abcnews.go.com/Health/story?id=4521341&page=1`

is an ABC news in March 2008 about an Oregon transgendered man claiming he was pregnant. The query and click log were collected around March 2008. Apparently, the users at that time period preferred the third document to the second one. But from click logs we notice that there are 521 sessions with at least one click on the second document and 340 sessions on the third one. If we only rely on click frequency, even after we discount the factor of click frequency difference caused by ranking positions at 2 and 3 (ref. Table 1), we may still be misled to an incorrect conclusion that the second document is more relevant than the third one. However, when we look into the data, we find that there are 266 sessions where the second document is clicked before the third one, while there are only 12 sessions in which a reversed click order are observed. This sequential click pattern clearly explains the "relevance disorder": Most of the time, the users who clicked the second document were not satisfied with the information they acquired, and proceeded to click the third one; however, if the users have clicked the third document, they then seldom need to click the second one, indicating the higher relevance of the third document than the second one.

Similar scenarios and sequential click patterns are also observed in many other aggregated sessions. The examples provided above are only to illustrate that certainly there are some sequential click patterns that are embedded in an aggregation of user clicks, and these click patterns provide substantial relevance information of the documents displayed in the search results. Due to the sequential nature of user clicks, a local model, which is defined on a single document, is therefore not capable of modeling user interactions with the search results. This motivates to use sequential models for click modeling that can take all the documents as its inputs and infer the relevance labels of all the documents *jointly*. Furthermore, in respect to the ranking nature of web search, we refer this ranking-targeted sequential learning as global ranking.

One important issue needs to be clarified here is that in our click modeling we are not using single user's click se-

Table 2: The click features used in the model.

Position	Position of the document in the result list
ClickRank	Rank of the 1st click of doc. in click seq.
Frequency	Average number of clicks for this document
FrequencyRank	Rank in the list sorted by num. of clicks
IsNextClicked	1 if next position is clicked, 0 otherwise
IsPreviousClicked	1 if previous position is clicked, 0 otherwise
IsAboveClicked	1 if there is a click above, 0 otherwise
IsBelowClicked	1 if there is a click below, 0 otherwise
ClickDuration	Time spent on the document

quence as an input to the global ranking, instead a sequence of *aggregated* click features (statistics) is used. This is because for a given query, generally, different users or even the same user at different time, may have different click sequences, and some are actually quite different from others; but over many user sessions, certain consistent patterns may emerge, and these are the basis for the click model we exploit to infer the relevance labels of the documents. In the next, we will discuss what kinds of click features are used in the model and how these aggregated click features are extracted from user click sessions.

4.1 Click Feature Extraction from Aggregated Sessions

The features that are used in our model are listed in Table 2. All these features are click-related and can be extracted from user clicks. Besides these features, no other textual features or hyperlink-related features are used in the model. Figure 2 illustrates the process of feature extraction from an aggregated session $<q$, 10-docs, an aggregation of user clicks$>$, where $\mathbf{x}^{(q)} = \{x_1^{(q)}, x_2^{(q)}, \dots, x_{10}^{(q)}\}$ denotes a sequence of feature vectors extracted from the aggregated session, with $x_i^{(q)}$ representing the feature vector extracted for document i. Specifically, to form feature vector $x_i^{(q)}$, first a feature vector $x_{i,j}^{(q)}$ is extracted from each user j's click information, and $j \in \{1, 2, \dots\}$, then $x_i^{(q)}$ is formed by averaging over $\{x_{i,j}^{(q)}, \forall j \in \{1, 2, \dots\}\}$, i.e., $x_i^{(q)}$ is actually an aggregated feature vector for document i. Note that some of the features in Table 2 are statistics independent of temporal information of the clicks, such as "Position" and "Frequency", but the other features are relying on their surrounding documents and the click sequences. Finally, for the purpose of training, each query-document pair is assigned a label by human judges, with $\mathbf{y}^{(q)} = \{y_1^{(q)}, y_2^{(q)}, \dots, y_{10}^{(q)}\}$ representing the sequence of assigned relevance labels.

From the experiments that follows, we find that "Frequency" is one of the most important features considered in the model. However, we should emphasize that the raw "Frequency" feature itself is quite noisy [10]; only when it is used jointly with other features, "Frequency" becomes a rather reliable indicator of the relevance labels. In addition, "Position" is another important feature considered in the paper since it indicates the label context produced by the baseline ranking models. Generally, the baseline ranking is imperfect, but it is not totally random: There is in general a trend that the top documents are more relevant than the bottom documents in the search results, and the algorithm should utilize this context information.

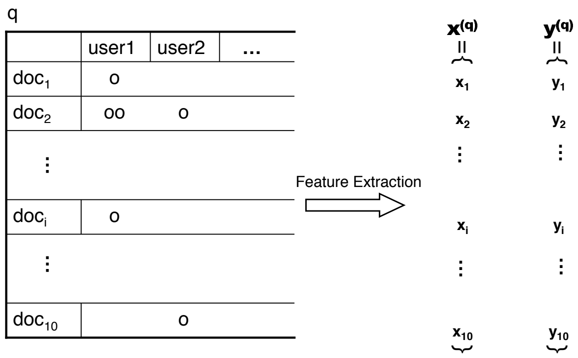

Figure 2: An illustration of feature extraction for an aggregated session. $\mathbf{x}^{(q)}$ denotes an extracted sequence of feature vectors, and $\mathbf{y}^{(q)}$ denotes the corresponding label sequence that is assigned by human judges for training.

5. THE GLOBAL RANKING ALGORITHMS

It is generally a challenging task to develop a global ranking algorithm, which can fully utilize all the local and global information among the documents to produce a document rank. Fortunately, under a loose approximation, the global ranking model defined in Equation (2) can be regarded as a sequential supervised learning problem [12, 5] since both have almost identical functional expressions. Therefore, the existing research in sequential supervised learning can be useful of solving the global ranking problem. One example of global ranking algorithm that follows in this direction is the one proposed in [14], in which the CRF algorithm is modified to handle continuous features and ranking scores. Without solving a ranking problem directly, however, this modified CRF algorithm is more in the sense of regression. In respect to the ranking nature of Web search, in this section we further explore several sequential supervised algorithms, including the CRF [12], the sliding window method and the recurrent sliding window method [5], to the global ranking problem. We emphasize the importance to adapt these algorithms to the ranking problem.

5.1 Conditional Random Fields

The conditional random fields [12] (CRFs) are one of the well-known probabilistic models for sequential labeling. Compared to the hidden Markov models [15] (HMMs), which define a joint probability distribution $p(\mathbf{x}, \mathbf{y})$ over an observation sequence $\mathbf{x}$ and a label sequence $\mathbf{y}$, the CRFs define a conditional probability distribution $p(\mathbf{y}|\mathbf{x})$ directly, which is used to label a sequence of observations $\mathbf{x}$ by selecting the label sequence $\mathbf{y}$ that maximizes the conditional probability. Because the CRF model is conditional, dependencies among the observations $\mathbf{x}$ do not need to be explicitly represented, affording the use of rich, global features of the input. Therefore, no effort is wasted on modeling the observations, and one is free from having to make unwarranted independence assumptions as required by the HMMs [12, 19, 21].

A CRF is simply a conditional distribution $p(\mathbf{y}|\mathbf{x})$ with an associated graphical structure, defining the dependencies among the components y_i of $\mathbf{y}$ (i.e., how to factorize $p(\mathbf{y}|\mathbf{x})$), globally conditioned on the observations $\mathbf{x}$. The simplest and most commonly used structure for modeling sequences is a linear chain, and the corresponding conditional distribution is defined as follows:

$$p(\mathbf{y}|\mathbf{x}) \propto \exp\left\{\sum_{j,t}\lambda_j f_j(y_t, y_{t-1}, \mathbf{x}) + \sum_{k,t}\mu_k g_k(y_t, \mathbf{x})\right\}, \quad (3)$$

where $f_j(y, y', \mathbf{x})$ is a transition feature function, $g_k(y, \mathbf{x})$ is an observation feature function, and

$$\Lambda = \{\lambda_1, \lambda_2, \cdots, \mu_1, \mu_2, \cdots\}$$

are the parameters to be estimated. In general, the feature functions in Equation (3) are defined on the *entire* observation sequence $\mathbf{x}$. However, in practice, due to computational issues and to avoid overfitting a subset of $\mathbf{x}$ is adopted in each feature function, and j and k in Equation (3) iterate over arbitrary subsets of $\mathbf{x}$, either in time dimension or in feature dimension.

Given i.i.d. training data $\mathcal{D} = \{\mathbf{x}^{(i)}, \mathbf{y}^{(i)}\}_{i=1}^{N}$, the maximum likelihood estimate can be used to compute the parameters Λ from

$$l(\Lambda) = \sum_{i=1}^{N} \log p(\mathbf{y}^{(i)}|\mathbf{x}^{(i)}), \quad (4)$$

which is a concave function and can be optimized efficiently by using the quasi-Newton methods, such as BFGS [19]. Once the parameters Λ are determined, given a new observation sequence $\mathbf{x}^*$, the most probable label sequence $\mathbf{y}^*$ can be computed by using the Viterbi algorithm [15].

Originally developed in computational linguistics and bioinformatics, the CRF feature functions used in Equation (3) are often in the form of the Kronecker delta function [12]. This choice of feature function results in a very efficient optimization method, but it also restricts the inputs and outputs of the CRF have to be discrete values, i.e., the label sequence $\mathbf{y}^*$ is a sequence of discrete values, each one corresponding to a relevance grade of one document. Although $\mathbf{y}^*$ computed from the Viterbi algorithm is the most probable one, from the experiments we find that it tends to have a majority of $y_i^* \in \mathbf{y}^*$ with the same labels. This is likely due to the limited label categories (i.e., 5 grades) compared with the relatively larger length of a label sequence (i.e., $T = 10$). Thus, a method that can produce continuous ranking scores is highly desired. We therefore use the following approximation for this purpose.

Besides generating the most probable label sequence $\mathbf{y}^*$, the Viterbi algorithm also yields the class probabilities for each label y_i in $\mathbf{y}$, i.e., $p(y_i = g|\mathbf{x}^*), \forall\, i \in \{1, 2, \cdots, T\}$ and $g \in \{0, 1, 2, 3, 4\}$, where g denotes a relevance grade, with $g = 4$ corresponding to `Perfect` and $g = 0$ to `Bad`, and so on. So, we may use the *expected relevance* to convert class probabilities into ranking scores:

$$\tilde{y}_i = \sum_{g=0}^{4} g \times p(y_i = g|\mathbf{x}^*). \quad (5)$$

Although Equation (5) is less of principle than the most probable sequence estimated from the Viterbi algorithm, in practice, we observe improved performance of this approximation over the Viterbi algorithm. In addition, the *expected relevance* (5) has been used in [13] to convert classification categories into soft ranking scores.

Note that the CRF algorithms discussed above and in [14] all tackle a ranking problem as a classification/regression problem since both optimize the CRF parameters in a maximum likelihood estimate without considering score ranks. It would be very challenging to adapt the CRF completely to a global ranking algorithm due to its complicated model assumptions. We leave this as an open question for our future study. Instead, we will explore two simplified sequential learning methods, such as the sliding window method and the recurrent sliding window method, and adapt them into the global ranking algorithms.

5.2 (Recurrent) Sliding Window Methods

The sliding window method converts the sequential supervised learning problem into an ordinary supervised learning problem [5]. In our ranking context, the scoring function f maps a set of consecutive observations in a window of width w into a ranking score. In particular, let $d = (w - 1)/2$ be the *half-width* of the window. The scoring function uses

$$\hat{x}_i = (x_{i-d}, x_{i-d+1}, \ldots, x_i, \ldots, x_{i+d-1}, x_{i+d})$$

as an *extended* feature to predict the ranking score $\hat{y}_i$, i.e., $\hat{y}_i = f(\hat{x}_i)$, $\forall i \in \{1, 2, \ldots, T\}$. Although this is a crude approximation to the CRF, the advantage of the sliding window method is its simplicity, which entails any classical ranking algorithms to be applied to the global ranking problem.

Similarly, in a recurrent sliding window method, the predicted scores of the *old* observations are combined with the extended feature to predict the score of the *current* observation. Particularly, when predicting the score for x_i, we can use the following available predicted scores $\hat{y}_{i-d}, \ldots, \hat{y}_{i-1}$ in addition to the sliding window to form the extended feature when predicting $\hat{y}_i$, i.e., the extended feature for x_i becomes

$$\hat{x}_i = (\hat{y}_{i-d}, \ldots, \hat{y}_{i-1}, x_{i-d}, x_{i-d+1}, \ldots, x_i, \ldots, x_{i+d}).$$

In contrast to the sliding window mehtod, the recurrent one can capture predictive information that was not being captured by the simple sliding window method. In our click modeling context, for example, if x_i is being clicked and x_{i-1} is not, then we probably should have $\hat{y}_i > \hat{y}_{i-1}$.

5.3 GBrank: A Boosting Algorithm for Preference Learning

As an advocated algorithm in this paper to implement the sliding window method and the recurrent sliding window method, GBrank [23] is discussed briefly in this section. Generally, GBrank is a learning to rank algorithm that is trained on preference data. It requires the training data be in the form of pairwise comparisons, i.e., one document is more relevant than another with respect to a query. Since in the training data, we have relevance grades assigned by human judges to each query-document pair, we can use these *absolute* relevance judgements to generate a set of preference data. For example, given a query q and two documents u and v, if u has a higher grade than v, e.g., `perfect` vs. `good`, we include the preference $u \succ v$ in the extracted preference set, and vice versa. This will be considered on all pairs of documents within a search session, except those with equal grades. By considering all the queries in the dataset, we therefore extract a set of preference data, denoted as

$$\mathcal{S} = \{\langle u_i, v_i\rangle \mid u_i \succ v_i, i = 1, 2, \ldots, M\}.$$

In [23], the problem of learning ranking functions is cast as the problem of computing a function h, such that h matches a given set of preferences as many as possible, i.e., $h(u_i) \geq h(v_i)$, if $u_i \succ v_i$, $i = 1, 2, \ldots, M$. The following objective function (squared hinge loss) is used to measure the risk of a given ranking function h,[2]

$$\mathcal{R}(h) = \frac{1}{2}\sum_{i=1}^{N}(\max\{0, h(v_i) - h(u_i) + \tau\})^2,$$

and we need to solve the following minimization problem

$$\min_{h \in \mathcal{H}} \mathcal{R}(h),$$

where $\mathcal{H}$ is a function class, chosen to be linear combinations of regression trees in our case. This minimization problem is solved by using functional gradient descent discussed in [7]. We summarize the GBrank algorithm [23] for learning ranking function h using gradient boosting as follows:

Algorithm GBrank:
Start with an initial guess h_0, for $k = 1, 2, \ldots, K$

1. Using h_{k-1} as the current approximation of h, we separate $\mathcal{S}$ into two disjoint sets,

$$\mathcal{S}^+ = \{\langle u_i, v_i\rangle \in \mathcal{S} | h_{k-1}(u_i) \geq h_{k-1}(v_i) + \tau\}$$

and

$$\mathcal{S}^- = \{\langle u_i, v_i\rangle \in \mathcal{S} | h_{k-1}(u_i) < h_{k-1}(v_i) + \tau\}$$

2. Fit a regression function (decision tree) $g_k(x)$ on the following training data

$$(u_i, \quad [h_{k-1}(v_i) - h_{k-1}(u_i) + \tau]),$$
$$(v_i, -[h_{k-1}(v_i) - h_{k-1}(u_i) + \tau]), \quad \forall \langle u_i, v_i\rangle \in \mathcal{S}^-$$

3. Form the new ranking function as

$$h_k(x) = h_{k-1}(x) + \eta g_k(x)$$

where η is a shrinkage factor.

Two parameters need to be determined: the shrinkage factor η and the number of iterations K, this is usually done by cross-validation.

6. EXPERIMENTS

We carry out an extensive experimental study of the proposed click modeling framework in this section. We first introduce the dataset used in our experiments, then describe the evaluation methods, followed by experimental results on performance comparison among different competitive methods.

6.1 Dataset

The user click data we used in the experiments are collected from a commercial search engine over a certain period of time. Following the procedures described in Section 4, we selected 9677 queries (and therefore 9677 aggregated sessions) from the user click logs that are both frequently queried by the users and have click rates over 1.0, where the click rate is defined as follows:

$$click\_rate(query) = \frac{\sum_{i \in sessions(query)} \#clicks(i)}{\#sessions(query)}. \qquad (6)$$

Such a selection of queries is to ensure that each aggregated session will have enough user clicks to accumulate statistically significant click features. We then requested human judges to label the top ten documents of each of 9677 queries to be `perfect, excellent, good, fair, or bad` according to their degrees of relevance with respect to the query. This constitutes our dataset, over which we examine the performance of the proposed click modeling methods.

6.2 Evaluation Metrics

Our evaluation focuses on the comparison between the predicted ranking of the click models and the original ranking produced by the baseline (production) models, i.e., the comparison in terms of search results re-ranking. We adopt the Discounted Cumulative Gain (DCG) criterion, a standard quality measure in information retrieval, to estimate the accuracy of various rankings. For example, the DCG score of a ranking is computed as

$$\mathrm{DCG}(L) = \sum_{i=1}^{L} \frac{2^{g(i)} - 1}{log(1+i)}, \qquad (7)$$

where L is the "truncation level" and is set to be $L = 5$ in our experiments, and $g(i) \in \{0, 1, 2, 3, 4\}$ is the relevance grade of the ith document in the ranked result set. $g(i) = 4$ corresponds to a "perfect" relevance, and $g(i) = 0$ corresponds to a "bad" relevance.

6.3 Performance Comparison

To illustrate the effectiveness of the supervised click modeling methods proposed in this paper, we test the performances of the CRF (Section 5.1), the sliding window method and the recurrent sliding window method (Section 5.2) on the collected dataset. As we have discussed in Section 5.2, the simplicity of the sliding window method and its recurrent version entails any ranking algorithms to be applied to the global ranking problem. We therefore implement the sliding window method and the recurrent sliding window method by SVM [20], GBDT [7] and GBrank [23], where SVM and GBDT are the regression algorithms but are used for the ranking problem, and GBrank is the only learning to rank method considered. As a comparison, we also examine the performance of the *cascade* model proposed in [4]. The measure of the performance is computed by the DCG(5) gains of the re-ranking over the original ranking from the baseline models.

Table 3 reports the performances among five different methods under different experiment settings, in which (1) CRF uses two observations (current and previous) in the definition of the observation function $g_k(y_t, x_t, x_{t-1})$, and the transition feature function $f_j(y_t, y_{t-1}, x_t, x_{t-1})$; this CRF configuration is selected because it yields the best performance over 10-fold CV; (2) the sliding window (SW) and the recurrent sliding window (RSW) methods as implemented by linear SVM, GBDT and GBrank with different window sizes, indicated by the digits following "SW" or "RSW". We have also included at the bottom of Table 3 two unsupervised methods: the *cascade* model, and a variant in which

[2] This loss function can be considered as a smooth surrogate of the total number of contradicting pairs in the given preference data with respect to the function h. We say $u \succ v$ is a contradicting pair with respect to h if $h(u) < h(v)$.

we add 1/position to the predicted scores. The idea is to bias the predictions towards the current ranking. We examine the performance of each method on different number of aggregated sessions filtered by four click rates: 1.0, 1.2, 1.5, and 1.7.

Table 3: The DCG(5) gains (%) of different algorithms on aggregated sessions extracted from click logs; the results are computed from 10-fold CV.

Click Rate		>1.0	>1.2	>1.5	>1.7
#sessions		9677	6795	2042	1119
CRF		0.91	1.15	2.07	2.50
SVM	SW1	-0.09	0.04	0.21	0.41
	SW2	0.39	0.38	0.32	1.26
	SW3	0.08	0.30	-0.06	1.06
GBDT	SW1	0.77	0.97	1.83	2.60
	SW2	1.01	1.26	1.98	2.79
	SW3	1.02	1.22	2.16	2.44
GBrank	SW1	0.77	1.00	2.13	3.03
	SW3	1.15	1.39	2.34	3.19
	SW5	1.17	1.48	2.35	3.40
	SW7	1.23	1.42	2.34	3.23
GBrank	RSW3	1.36	1.67	2.62	3.93
	RSW5	1.59	2.02	**3.07**	**4.06**
	RSW7	**1.67**	**2.11**	3.04	3.64
Cascade	w/o pos.	-3.57	-3.62	-1.84	-0.20
	with pos.	1.26	1.49	2.44	3.33

It is demonstrated in Table 3 that (1) As the click rate increases, all methods in general have increasing DCG(5) gains over the original rankings from the baseline models. Since our supervised click modeling methods only exploit user click information, it is expectable that the more clicks in a user session, the more information can be exploited by the model for reliable prediction. (2) Better than the baseline models, CRF does not show the best performance among different competitive methods. This is likely due to the restricted modeling assumptions and the regression nature of the algorithm. (3) The tree-based methods, such as GBDT and GBrank, outperforms the linear SVM. This is likely because a tree-based model is in general more expressive than a linear model. (4) As the window size increases, in general, the sliding window (SW) methods and the recurrent sliding window (RSW) methods have the increasing DCG(5) gains, demonstrating that neighboring observations indeed carrying useful information for ranking score predictions. (5) The SVM and GBDT with the recurrent sliding windows have significant dropped DCG(5) gains (about -6%) over the original rankings. (For concise, these results are not provided in the table.) This is because SVM and GBDT explicitly deal with a ranking problem as a regression problem. As we noticed from the experiments, although the recurrent SVM and GBDT have smaller regression errors than their sliding window implementations, the corresponding DCG(5) gains are indeed much worse. This clearly demonstrates the discrepancies between the objectives of regression algorithms and ranking algorithms. (6) Among all the algorithms considered, GBrank(RSW) outperforms all the other methods, demonstrating the pair-wise ranking methods, such as GBrank, are more suitable to our click modeling framework. (7) The original cascade model (w/o pos.) underperforms the baseline models, but a variant of it (with pos.) dramatically improves its performance. As we have discussed in Section 4.1, the label context information as indicated by "Position" is one of the important features in the click data modeling, and an algorithm probably should use this information if available.

6.4 Comparisons to The Heuristic Rule based Methods

With the ranking scores predicted by the supervised click modeling methods, we can extract preference pairs in the form of $(u \succ v)$, which represents that document u is more relevant than document v with respect to a query. This kind of preference pairs can also be extracted via the heuristic rule based methods, such as SkipAbove and SkipNext [16]. This raises the question about which method is more accurate in extracting preference pairs, assuming that the ground truth is the preference extracted from relevance grades assigned by human judges. Since the experiments in Section 6.3 demonstrate that GBrank(RSW7) has the best performance over all the other methods considered, in this part of experiments, we only consider the performance comparison between GBrank(RSW7) and the heuristic rule based methods.

Both SkipAbove and SkipNext are the heuristics that are derived from eyetracking studies [16]. Specifically, the first strategy, SkipAbove proposes that given a clicked-on document, any higher ranked document that was not clicked on is less relevant, while the second strategy, SkipNext claims that for two adjacent documents in the search results if the first document is clicked on, but the second is not, the first is likely more relevant than the second. Since SkipAbove extracts the preference pairs in a reverse order of the original rankings, accurately extracted SkipAbove pairs are more precious than those extracted from SkipNext, due to their potentials of correcting the baseline ranking models.

To compare the performance of GBrank(RSW7) against SkipAbove and SkipNext, we separate the GBrank(RSW7) extracted pairs into two categories: "SN Pairs" and "SA Pairs", according to whether the extracted pairs are consistent with the original rankings or not. The category of "SN Pairs" is consistent with the original rankings, and is compared with SkipNext; and the other category "SA Pairs" is compared with SkipAbove. For each extracted preference pair, there are three cases that can occur when it is compared with the ground truth: (1) agree with ground truth, (2) disagree with ground truth due to tied preference, and (3) disagree with ground truth due to opposite preference. By adjusting the thresholds in extracting pairs from GBrank(RSW7), SkipAbove and SkipNext, we generate approximately equal number of preference pairs from each method, and compute the percentages for each of the three occurrences, with the results reported in Table 4.

It is demonstrated in Table 4 that GBrank(RSW7) outperforms both SkipAbove and SkipNext by a large margin on the dataset considered. In particular, GBrank has 87.84% accuracy in generating SkipNext-like preference pairs, and 41.7% accuracy in generating SkipAbove-like perference pairs, while SkipNext and SkipAbove only have 55.23% and 27.87% accuracy, respectively. This by a large part is contributed by the supervised learning properties of our click modeling methods, while SkipAbove and SkipNext are, in some

Table 4: The performance comparison of GBrank(RSW7), SkipAbove and SkipNext on 9677 aggregated sessions extracted from click logs; the GBrank results are computed from 10-fold CV.

Methods	#Pairs	Agree	Tie	Disagree
GBrank(SN Pairs)	16000	**87.84**	9.17	2.99
SkipNext	16000	55.23	32.35	12.42
GBrank(SA Pairs)	1000	**41.71**	39.48	18.81
SkipAbove	1000	27.87	37.88	34.25

sense, unsupervised learning methods. In addition, an interesting fact worth of emphases is the large discrepancy on the number of preference pairs generated from GBrank(SN Pairs) and GBrank(SA Pairs). Indeed, given a high quality baseline ranking model, it is much more difficult to generate accurate SkipAbove pairs than their SkipNext counterparts.

7. CONCLUDING REMARKS

This paper focuses on extracting relevance information from user click data via a global ranking framework, and global ranking is explored here to utilize the relational information among the documents as manifested by an aggregation of user clicks. Experiments on the click data collected from a commercial search engine demonstrate the effectiveness of the proposed method and its superior performance over a set of widely used unsupervised methods, such as the cascade model [4] and the heuristic rule based methods [16]. Although it is unfair to compare a supervised method with an unsupervised method, this paper introduced an approach in cases some human judgements are available for information extraction. We believe that the supervised approach can be more reliable than the unsupervised approach because user click data are inherently very noisy; by exploring supervised learning in click data modeling, we expect our click model can reliably extract relevance information by calibrating with human relevance judgments. Some future research directions that are worth of further investigation include: (1) a CRF algorithm that is fully adapted to the global ranking problem; and (2) exploring other supervised learning algorithms, such as those proposed in the context of structured learning [11], to the global ranking problem.

8. REFERENCES

[1] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. In *Proceedings of the seventh international conference on World Wide Web*, 1998.

[2] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender. Learning to rank using gradient descent. In *Proceedings of the 22nd International Conference on Machine Learning*, 2005.

[3] B. Carterette and R. Jones. Evaluating search engines by modeling the relationship between relevance and clicks. In *NIPS*, 2007.

[4] N. Craswell, O. Zoeterm, 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)*, 2008.

[5] T. G. Dietterich. Machine learning for sequential data: a review. *Lecture Notes in Computer Science*, (2396):15–30, 2002.

[6] G. 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)*, 2008.

[7] J. Friedman. Greedy function approximation: a gradient boosting machine. *Ann. Statist.*, 29:1189–1232, 2001.

[8] N. Fuhr and C. Buckley. A probabilistic learning approach for document indexing. *ACM Transactions on Information Systems*, 9:223–248, 1991.

[9] T. Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the ACM Conference on Knowledge Discovery and Data Mining (KDD)*, 2002.

[10] T. Joachims, L. Granka, and B. Pan. Accurately interpreting clickthrough data as implicit feedback. In *Proceedings of 28th ACM SIGIR*, 2005.

[11] A. Kulesza and F. Pereira. Structured learning with approximate inference. In *NIPS*, 2007.

[12] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, pages 282–289, 2001.

[13] P. Li, C. Burges, and Q. Wu. Mcrank: Learning to rank using multiple classifications and gradient boosting. In *NIPS*, 2008.

[14] T. Qin, T. Liu, X. Zhang, D. Wang, and H. Li. Global ranking using continuous conditional random fields. In *NIPS*, 2008.

[15] L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. In *Proceedings of the IEEE*, pages 257–286, 1989.

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

[17] F. Radlinski and T. Joachims. Active exploration for learning ranking from clickthrough data. In *Proceedings of the ACM International Conference on Knowledge Discovery and Data Mining (KDD)*, 2007.

[18] G. Salton. *Automatic Text Processing.* Addison Wesley, MA, 1989.

[19] C. Sutton and A. McCallum. *An introduction to conditional random fields for relational learning*, chapter Book chapter in Introduction to Statistical Relational Learning. MIT Press, 2006.

[20] V. N. Vapnik. *The Nature of Statistical Learning Theory.* Springer, 1995.

[21] H. M. Wallach. Conditional random fields: An introduction. Technical report, Dept. of Computer and Information Science, University of Pennsylvania, 2004.

[22] X. Wang and C. Zhai. Learn from web search logs to organize search results. In *Proceedings of the 30th ACM SIGIR*, 2007.

[23] Z. Zheng, H. Zha, K. Chen, and G. Sun. A regression framework for learning ranking functions using relative relevance judgments. In *Proceedings of the 30th ACM SIGIR conference*, 2007.