A Model to Estimate Intrinsic Document Relevance from the Clickthrough Logs of a Web Search Engine

Georges Dupret
Yahoo! Labs
gdupret@yahoo-inc.com

Ciya Liao
Yahoo! Labs
ciyaliao@yahoo-inc.com

ABSTRACT

We propose a new model to interpret the clickthrough logs of a web search engine. This model is based on explicit assumptions on the user behavior. In particular, we draw conclusions on a document relevance by observing the user behavior after he examined the document and not based on whether a user clicks or not a document url. This results in a model based on intrinsic relevance, as opposed to perceived relevance. We use the model to predict document relevance and then use this as feature for a "Learning to Rank" machine learning algorithm. Comparing the ranking functions obtained by training the algorithm with and without the new feature we observe surprisingly good results. This is particularly notable given that the baseline we use is the heavily optimized ranking function of a leading commercial search engine. A deeper analysis shows that the new feature is particularly helpful for non navigational queries and queries with a large abandonment rate or a large average number of queries per session. This is important because these types of query is considered to be the most difficult to solve.

Categories and Subject Descriptors

H.4 [**Information Systems**]: Miscellaneous

General Terms

Algorithms

Keywords

Clickthrough Data, User Behavior Model, Search Engines

Introduction

Web search engines clickthrough logs are a large and important source of information about user behavior. This feedback provides detailed and valuable information about users interactions with the system as the issued query, the presented URLs, the selected documents and their ranking. It is a poll of millions of users over an enormous variety of topics. It has been used in many ways to mine user interests and preferences. Examples of applications include Web personalization, Web spam detection, query term recommendation. Unlike human tags and bookmarks, implicit feedback is not biased towards "socially active" Web users. That is, the data is collected from all users, not just users that choose to edit a wiki page, or join a social network such as MySpace, Friendster or Twitter.

Click data seems the perfect source of information when deciding which documents (or ads) to show in answer to a query. It can be thought as the result of users voting in favor of the documents they find interesting. This information can be fed back into the engine, to tune search parameters or even used as direct evidence to influence ranking [2, 17]. Nevertheless, clickthrough rates on a document cannot be understood as a direct measure of relevance for a variety of reasons. Besides spammers who intentionally advertise one content and deliver another, not all document snippets reflect accurately the document content. Other presentation bias have a strong influence on user clicks. The position of the document in the ranking for example heavily skew user decisions [19]. Experiments also show that a document is not clicked with the same frequency if situated after a highly relevant or a mediocre document.

We can divide in essentially four categories according to how click information has been used in the Literature:

1. Works where the objective is to gain insight into typical user behavior [23, 8, 11, 18],

2. Estimate of the document relevance are deduced from the user interactions with the search engine. This work falls into this category. These estimates can be used either as features for a ranking algorithm, or as a target to augment a set of editorial judgments with more data. [1, 17, 24, 6, 10],

3. The click patterns are interpreted in order to compare different ranking functions, i.e. to derive a metric [5, 16, 18, 9],

4. Clicks have been used to directly re-rank the top documents retrieved from search engine in [15].

In general, the hope is that an appropriate use of click data should help the search engine to adapt better to the user needs than editorial evaluations for example.

WSDM'10, February 4–6, 2010, New York City, New York, USA.

Contributions and Organization.

We propose a model of user behavior based on the user actions *after* he has consulted a document, i.e. after he has clicked an url and examined the corresponding document[1]. The objective of the model is to derive an estimate of document intrinsic relevance, as opposed to perceived relevance or attractiveness. In Section 1 we describe the model. In Section 2 we compare the hypothesis and the performance of this model to two existing and related models. In Section 3.2 we evaluate experimentally the usefulness of the model estimates as a feature for a ranking algorithm. We conclude that the feature is particularly helpful for queries with a large abandonment rate or the informational queries with a large average number of click per session.

1. SESSION UTILITY MODEL

The model we propose here makes the assumption that users search and click on document until they satisfy their information need. Given a set of document, we will attempt to predict whether this set of documents contains enough information for the user. This prediction will be based on the behavior of previous users for the same query. We first introduce some definitions and notations.

Different definitions of user session co-existing in the Literature. We start by stating the one we use here. A session is the set of actions a user undertakes to satisfy a given information need. As such, a session may include several search results page views and query reformulations. The user may follows several query suggestions in the same the session.

We denote the set of queries, issued by the user or resulting from the user accepting a query suggestion, by q_0, q_1, etc., with the subscript reflecting the order in which queries are submitted to the engine.

The basic intuition at the origin of the model we propose here is that the user searches for results until he gathers enough information to satisfy his need, at which time he stops the search. Each clicked document brings a certain amount of information that the user accumulates with the information provided by the documents clicked previously. To simplify the discussion, we define a prototypical session:

1. The user issues a query q_0,
2. He clicks on two documents d_3 and d_5, in that order,
3. He reformulates his query to q_1,
4. He clicks on the document d_{13} in the new ranking,
5. He finishes the search.

This model requires to identify the user sessions from the query logs. Typically, the logs contain a unique identifier or cookie associated to the user, a time when the query is issued and the time when the documents are clicked along with the query string and the urls. On the other hand, there is no specific information on whether two consecutive queries issued by a given user belong to the same session. User session identification is an on-going problem and some work has been done already to address it [3, 22]. Results are encouraging and for the purpose of this work we will assume that a reliable method exists.

[1]Studies on user behavior after the user has read the document is common in interactive information retrieval [21] but to our knowledge this is the first work to apply this to web logs analysis.

1.1 Assumptions

The central assumption in this model is that the user searches the result list of the original query and its reformulations until he has gathered enough information to satisfy his need, independently of how far the relevant documents are in the ranking or how hard it is to come up with a good reformulation. All sessions ending with a click are therefore considered as successful. There is no notion of a user abandoning a search out of despair in this model. We are not arguing that this assumption is verified in reality –it is not– but numerical experiments will show that the resulting model is nevertheless useful.

Each clicked document d_r provides some utility u_r to the user. We make the hypothesis that utilities are additive: The utility of the prototypical session at the end of the search is $U(\mathcal{C}) = \sum_{r\in\mathcal{C}} u_r = u_3 + u_5 + u_{13}$ where $\mathcal{C} = \{d_3, d_5, d_{13}\}$ is the set of clicked documents. The assumption that utilities can be simply added is clearly an approximation. More realistically the total utility of a set of documents is lower than the sum of individual document utilities because some documents might partially or fully repeat the same information.

After the user clicked on document d_3 of our example, she acquired a quantity u_3 of utility. The fact that she continues her search after consulting the document indicates that she did not obtain enough utility to satisfy her information need. After consulting document u_5, she acquired an amount $u_3 + u_5$ of utility, but still not enough for her to stop. Only after consulting $u_3 + u_5 + u_{13}$ does she have enough information and she stops the search. The different steps of this search are summarized in the first three columns of Table 1.

Sometimes a user clicks several time on the same document. If the time between two clicks is small, and if no other document has been clicked in between, then this might denote either that the user is used to double-clicking, or that the network latency is large. In this case, repeated clicks can be treated as a single click. On the other hand, if the time lapse between two clicks on the same url is large or the user clicked other documents in between, this might be that the user came to the conclusion that the document he visited multiple times in the same session is probably one of the best documents he can get. We chose in this work to ignore the sessions with multiple clicks on the same document because it is easier, and because we dispose of a large amount of clickthrough logs. Nevertheless a more careful analysis might reveal that this type of session is particularly informative.

1.2 Description of the Model

The variable the model attempt to predict is whether, given a certain amount of utility, the user will stop or continue her search. We associate a binary variable s to this elementary event. It is equal to 1 if the user stops and 0 if he continues the search.

Event Likelihood.

After clicking on a set of document $\mathcal{C}$ the amount of utility the user gathered is $U(\mathcal{C})$, a value between 0 and infinity. We assume that the user stops with a probability that depends monotonely on this amount. This suggests the use of

Table 1: Example Session. The first column represents the document in the order they have been clicked. The second column is the amount of utility the user gathered *after* clicking them. The third column reports whether the click is the last action of the user session. Finally, the last column reports the probability –according to the model– of the event reported in the previous column.

document	utility	stop	event probability
d_3	u_3	0	$1-\sigma(u_0+u_3)$
d_5	u_3+u_5	0	$1-\sigma(u_0+u_3+u_5)$
d_{13}	$u_3+u_5+u_{13}$	1	$\sigma(u_0+u_3+u_5+u_{13})$

a logistic function to map $U(\mathcal{C})$ to a probability of stopping:

$$P(s=1|U(\mathcal{C})) = \sigma(U(\mathcal{C})) = \sigma(u_0+\sum_{r\in\mathcal{C}} u_r) \quad (1)$$

$$= (1+\exp(-u_0-\sum_{r\in\mathcal{C}} u_r))^{-1} \quad (2)$$

- $\sigma(x)$ is the logistic function:

$$\sigma(x) = \frac{1}{1+e^{-x}}$$

- $\mathcal{C}$ is a set of clicked documents,
- $U(\mathcal{C})=\sum_{r\in\mathcal{C}} u_r$ is the sum of the utilities of the documents in $\mathcal{C}$ and,
- u_0 is a query dependent intercept.

The likelihoods of the three events in our example session are reported in Table 1.

Session Likelihood.

The likelihood of a session is simply the product of the likelihood of the events belonging to that session. The join likelihood of the prototypical session in our example can be written:

$$\begin{aligned} L_s &= P(s=0|d_3)P(s=0|d_3,d_5)P(s=1|d_3,d_5,d_{13}) \\ &= (1-(1+e^{-(u_0+u_3)})^{-1}) \\ &\quad \times(1-(1+e^{-(u_0+u_3+u_5)})^{-1}) \\ &\quad \times(1+e^{-(u_0+u_3+u_5+u_{13})})^{-1} \end{aligned}$$

We can write the likelihood as a product because, given the sets of clicked documents, the probabilities of stopping are independent.

Query Likelihood.

The join likelihood L_{q_0} of the sessions of a query is the product of the likelihood of all its sessions.

Utility Estimation.

To obtain an estimate of the individual document utilities $\{u\}$ and the intercept u_0 we maximize the join likelihood L_{q_0} with respect to the utilities and the intercept. Because click logs tend to be sparse and noisy, it is generally a good idea to introduce a prior on the document utilities and compute the *Maximum a Posteriori* (MAP) instead of the maximum likelihood estimate:

Table 2: Predicting session end as a probabilistic binary classification problem. Each row in this table correspond to a training example deduced from the corresponding row in Table 1. The first column corresponds to the intercept, the last to the target.

u_0	u_1	u_2	u_3	u_4	u_5	...	u_{13}	...	target
1	0	0	1	0	0	...	0	...	0
1	0	0	1	0	1	...	0	...	0
1	0	0	1	0	1	...	1	...	1

$$\text{MAP} = \arg\max_{\{u\},u_0} \left(L_{q_0} \times \prod_{u_i\in\{u\}} \mathcal{N}(u_i;u_p,\sigma_p) \right) \quad (3)$$

where we choose a normal prior $\mathcal{N}(u_i;u_p,\sigma_p)$ centered around a value u_p with a variance σ_p^2 to be adjusted by cross-validation[2]. We could choose other prior. In particular, we could impose that utilities be positive by using a log-normal distribution instead of a normal. In the current work, the mean prior utility of a document u_p and its variance σ_p^2 are taken to be identical for all documents and all queries and are adjusted by cross-validation on the training set. In fact, nothing prevent the use of different values at the query or the document level to reflect prior knowledge. For example, if we have a relevance score for each document, we can choose an individual document prior utility to reflect this score.

For practical reasons, the logarithm of the MAP is usually maximized instead of the MAP itself.

Document Set Relevance.

The problem of predicting user stops can be seen alternatively as a binary classification problem where the feature vector entries correspond to the documents and are set to 1 if the document is clicked, 0 otherwise. The target is then set to 0 if the user continued the search after consulting the documents indicated by a one in the feature vector, 1 if the user stopped. This is represented for the three events of our example in Table 2. Depending on the classifier the utilities can or cannot be available after training. This suggests the following definition:

DEFINITION 1 (DOCUMENT SET RELEVANCE). *The relevance of a set of document $\mathcal{C}$ to a query q_0 is the probability that a user stops after clicking it.*

By extension, the relevance of a document is the relevance of the set containing only that document. The relevance of a particular document is then the output of the probabilistic classifier when the input vector is 0 everywhere but for u_0 and the entry corresponding to the document.

Note that the logarithm of the MAP in Eq.3 can be understood as the negative of the loss function of a Logistic Regression[3]. Call X the binary feature vector over the clicked

[2]the variance and logistic function symbols σ can be distinguished from the context.

[3]Note that Logistic Regression has been used before in the context of clickthrough logs to predict clicks [7, 10] or for evaluation [5]. These works differ significantly from this one by the assumptions and the variables they use.

documents (one row of Table 2, without the target); Call U the corresponding vector of utilities (including the intercept u_0). The utility corresponding to a sequence of clicks defined by X is $X^T U$ and the probability of ending the search is $\sigma(X^T U)$. A possible loss function over this observation is:

$$-\log\ \sigma(X^T U)^s - \log(1-\sigma(X^T U))^{1-s}$$

where $s = 1$ if the click sequence leads to a stop. If we sum the losses of all event and add a regularization term for each utility we obtain minus the logarithm of Eq. 3. The problems of maximizing the MAP and minimizing the loss are thus equivalent. If we use a logistic regression as a classifier the relation between relevance and utility is simply:

$$P(s=1|u_i) \quad = \quad \sigma(u_o + u_i)$$

Ranking.

We can order documents indifferently according to their relevance or their utility. Both lead to the same ranking. The model proposed here doesn't predict clicks and consequently is not a complete generative model. Strictly speaking we cannot deduce an optimal ranking from it. If we make the extra assumption that users click on a document with a probability proportional to the document utility, then ordering documents according to utility minimizes the search length.

Generalization.

We remark that we don't need to assume that the user browses the result list sequentially. The important information to maintain is the chronological order of the clicks. This suggests the following, more formal, reformulation. Be T^n the number of documents clicked during session n of query q and $\mathcal{C}_t^n$ the set composed of the first t documents clicked during that session. The utility achieved by the user is $U(\mathcal{C}_t^n) = \sum_{i=1}^{i=t} u_i$. Be s_t^n the binary variable that is *true* or 1 if the user stops the search after consulting the t^{th} document. We have $P(s_t^n = 1|U(\mathcal{C}_t^n)) = \sigma(\sum_{i=0}^{i=t} u_i)$. The likelihood of the query is written:

$$L_q \quad = \quad \prod_{n=1}^{N} \prod_{t=0}^{t=T^n} (\sigma(\sum_{i=0}^{t} u_i))^{s_t^n} (1-\sigma(\sum_{i=0}^{t} u_i))^{1-s_t^n}$$

If $\mathcal{D}$ is the set of documents that have been clicked at least once during all the sessions of q, the evaluation problem reduces to the problem of maximizing $L_q \times \prod_{d\in\mathcal{D}} \mathcal{N}(u_d|u_p, \sigma_p)$ where, as before, $\mathcal{N}(u_d|u_p, \sigma_p)$ is the normal prior over document d, centered on u_p with a variance σ_p^2.

Discussion.

This work differs significantly from what is found in the Literature. To start with, the proposed model doesn't predict clicks; the set of clicked documents is an input to the model. This is therefore not a complete generative model. It also completely ignores documents that are not clicked and gives no relevance estimate for such documents. This is consistent with the idea that the true relevance of a document can be decided upon only after the document has been consulted. By contrast, in most models it is argued that the perceived relevance is aligned with the actual relevance (see Section 2).

Another notable characteristic is that the model doesn't take document positions into account, but only the sequence of clicks. It doesn't matter for the model if a document was clicked high or low in the ranking. This might seem counter-intuitive, but again it is a consequence of the fact that the true relevance of a document can only be assessed after it was seen. Once it has been seen, this relevance is independent of where the document was in the ranking.

The predicted relevance of a document is independent of its popularity. For example, the predicted relevance of the url `answers.yahoo.com` for the query `yahoo` is close to 1 (i.e. the document at `answers.yahoo.com` is almost perfectly relevant, actually as relevant as the url `www.yahoo.com`) even though it is rarely clicked. The reason is that some users issue the query `yahoo` with the intention to navigate to `answers.yahoo.com`. For these users, the yahoo answer page is perfectly relevant. Conversely, the model predicts a low relevance for some highly popular documents. A document that is span for example might catch the attention of many users, but as it doesn't contribute to satisfy the user information need, the user utility remains constant and probability of ending the search is not modified. In other words, the document doesn't affect the subsequent user behavior. The model is therefore expected to be robust to spam documents (but not to click spam).

While predicting a high relevance for a non popular document is fine per se. it might become problematic when ranking the documents according to relevance. This problem can be mitigated by choosing an appropriate prior: If the document prior parameters u_p and σ_p are chosen to be small, document utilities will tend to be small as well unless there is enough evidence to compensate the prior. This can only happen when the document receives many clicks; i.e. when it is popular.

The strongest assumption we made is clearly that users stops when their information need is met. Clearly, this need not be true. Users can also give up a search or decide to switch to another search engine. We could use some classifier to predict session success (see [14] for a work on this topic) and choose the queries to include accordingly. We can also remove from the queries with a large abandonment rate. The ultimate test about this model is whether it is useful in predicting document relevance, a problem to which we turn in the numerical experiments of Section 3.

In conclusion, the model sometimes predicts a document relevance totally in opposition to what the clickthrough rate or the perceived relevance would suggest. This is clearly a desirable property because it is robust to whether the snippet actually reflects the document relevance or not as for example in the case of spam. On the other hand, this can lead to problems when a document is highly relevant to only a few users, unrepresentative of the whole population. This can be partially at least compensated by a adequate prior.

2. RELATED WORKS

To our knowledge, the first attempt to explicitly model user behavior on a page of search results from the web logs is the Examination model described in [10] (A very similar model is described in [12]). In that work, users search the list of results sequentially and click on a document url if its snippet is (1) *examined*, an event that depends on the position in the ranking and the distance to the last click, and (2) is *attractive*. Attractiveness is a property of the document snippet, sometimes referred to as "perceived relevance". The model makes the assumption that attractive-

ness and relevance tend to be equal. Naturally, this need not be always the case but search engines make considerable effort so that they are aligned.

The user successive decisions are symbolized in Fig. 1. The user starts at the first rank –which he always examines– and then decides whether to click on the link depending on the document attractiveness. He then proceed to the next position in the ranking with a probability that depends on the position in the ranking and the distance to the previous click. For example, if the user made a click at position 2 and skipped position 4 and 5, the probability he will examine position 6 is a parameter $\gamma_{\text{rank=6,distance=4}}$ where $4 = 6 - 2$ is the distance to the last click. This parameters are evaluated by maximizing the likelihood of the observed sessions.

In the Examination model, the actual content of the documents clicked by the user have no influence on his decision to proceed with the search (Only the position of the last click has an influence). The models in [6, 13] attempt to address that problem. Both models are fairly similar and discuss the first in more details. A latent variable called *satisfaction* is meant at modeling whether the user was happy with the actual content of the document he clicked. The assumption is that if the document is satisfactory, then the user ends the search.

The decision process modeled by the Satisfaction model is represented in Fig. 2. The first three decision blocks are identical to the previous Examination model, but a new variable is introduced: If the user is satisfied with the document, then he ends the search with a high probability. If he is not satisfied he continues the search to the next position in the ranking with a high probability.

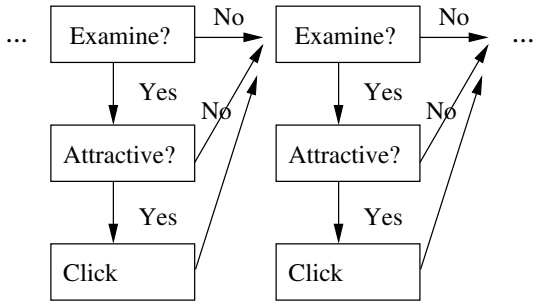

Figure 1: Examination Model Decision Graph

We see that, according to the taxonomy defined by Broder [4], the user of this model is engaged in a "navigational" search, i.e. the user is searching a particular piece of information that fit in a single page, or he is searching for a given web page he knows or assumes exists. A large proportion of searches fall into this category. If for example the probability of ending the search after finding a satisfactory document is set to 1, the satisfaction probability of a document becomes the proportion of the number of times the document was the last click of the session divided by the number of times the document was clicked. Attractiveness on the other hand is the number of time the document was clicked divided by the number of time the document was situated before or at the rank of the last click of a session.

It is interesting to compare these two models with the Session Utility model in terms of the assumptions they make. The emphasis in the Examination model is on predicting clicks and in particular on the influence of the position in the ranking. Arguably, by making the decision to continue the search depend on the rank and the distance, the model capture the influence of the position in the ranking: If the user is low in the ranking, then her propensity to abandon the search, and hence not to click anymore, is higher. Some notion of effort is also present in the idea that the number of previous clicks should influence the user subsequent behavior.

On the other hand, in the Satisfaction model, the decision to stop the search is based on whether the last clicked document fulfilled the user information he needed or not. The effort is also taken into account: The user has a certain probability, typically set to less than one, of examining the next document snippet whether he clicked or not at the previous step. As a consequence if the user never clicks, his probability of ending the search decreases exponentially.

By contrast, the Session Utility model is not attempting to model clicks and doesn't attempt to determine the influence of the position of the user on the search result page; It makes the assumption that users will make enough effort to obtain the results they need. On the other hand, unlike the Satisfaction model, it takes the utility of all documents into account to explain a session end; By doing so, the model has the potential to handle navigational and non navigational queries. A major drawback of the two Satisfaction models is that if a user decides to continue his search after the first click, he resumes his search as if nothing happened before. If the user clicked on position 5, for example, and did not end his search, these models predict that his behavior would be exactly equal as that of another user who would start his search at position 6. This is clearly not realistic, but because the vast majority of sessions consist of one click only, these models appear to perform well nevertheless.

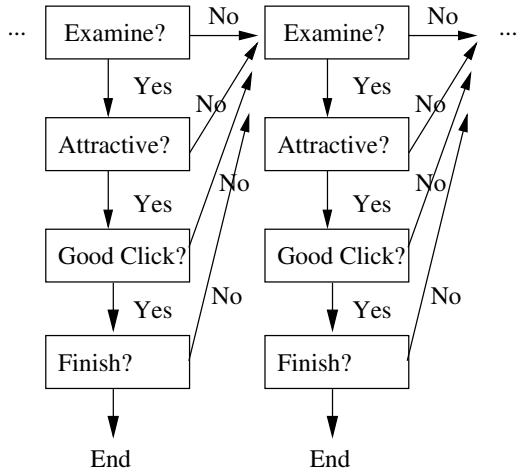

Figure 2: Satisfaction Model Decision Graph

3. NUMERICAL EXPERIMENTS

In this section we first compare the performances of the Examination the Satisfaction and the Session Utility models and discuss their differences. We then use the *relevance* estimates of the Session Utility model as a feature for a Machine Learned Ranking algorithm to test whether the feature can help optimize a leading commercial search engine.

3.1 Click Models Comparison

We first use one month of query logs from UK & Ireland and evaluate the Examination, the Satisfaction and the Session Utility models' parameters. We then select the query url pairs for which we dispose of an editorial relevance judgment on a five grade scales: `BAD`, `FAIR`, `GOOD`, `EXCELLENT`, `PERFECT`. We merge the `BAD` and `FAIR` labels into a common `BADFAIR` class. The meaning of these labels is straightforward, but it is important to know that a document can be labeled perfect if it is the target of a navigational query. Informational queries cannot contain `PERFECT` documents.

We then proceed as follows:

1. We compute the score differences between the documents of a given query for all queries. If the difference is positive, we conclude that the model predicted that the first document is more relevant than the second, or, in other words, that the first document is preferred to the second. We obtain a preference score for each document pair.

2. We order the pairs by decreasing absolute difference of their scores,

3. We use the editorial labels to derive a preference among the same document pairs, i.e. if the editorial label of the first document is superior to the label of the second document, then we say that the first document was preferred by the editors over the second document. We remove the ties from the set of pairs.

4. For a given proportion of the pairs –say 50%–, we compute the proportion among those pairs for which the preference predicted by the model and by the editors are in agreement.

The result are reported in Fig. 3 and correspond to the three "All" curves in black. For example, on the graph of the Satisfaction model, we observe that if we take 20% of the pairs with the largest *satisfaction* difference, close to 94% of these pairs are in agreement with the editorial judgments. Similarly, the 20% pairs with the largest difference in *relevance* score (Session Utility model) have 90% chance of being in agreement with the editorial judgment.

The data can be further divided into classes of preferences: `PERFECT` vs. `GOOD` or `EXCELLENT` vs. `BADFAIR`, etc. We see for example that if we take 80% of the data, the Session Utility model classify correctly 80% of the `EXCELLENT` vs. `BADFAIR` pairs.

The advantage of this evaluation method is that it doesn't need an algorithm to classify document scores to classes, which can lead to issues on which algorithm to choose and whether a particular algorithm is more favorable for a particular model. This evaluation method is also relatively insensitive to the scale of score differences because on the x-axis we report a proportion of the data set, not an difference between scores. The actual score difference for the successive percentiles of the data is reported in Fig. 3 as the (red) dashed line with the strongest negative slope.

Some conclusions can be drawn at first glance. The Session Utility and the Satisfaction models have a relatively similar performance and both offer better preference predictions than the Examination model. The Satisfaction model outperforms on average the Session Utility model when the score differences are relatively large but the Session Utility model catches up when all the preference relations are taken into account. It is interesting to observe the relative performance of the two models for the different classes of preferences. `PERFECT` vs. `BADFAIR` is the easiest class to predict, followed closely by `PERFECT` vs. `GOOD`. The Satisfaction model is better at distinguishing `PERFECT` vs. `EXCELLENT` pairs, while the comparative strength of the Session Utility model is at distinguishing `EXCELLENT` from `BADFAIR` documents. Overall, the Satisfaction model is good at identifying `PERFECT` documents; This can be explained by the fact that this model is biased towards navigational queries. The Session Utility model on the other hand is better at distinguishing the other classes of preferences.

Overall, the performance at predicting preference judgments of the two best models are comparable. The relative strength and weaknesses of the models reflect the assumption made on the user behavior: For the Satisfaction model, the user decision to continue a search after a click depends exclusively on the clicked document. This makes the model sensitive to `PERFECT` documents. The user of the Session Utility model decides to continue a search depending on the set of documents he has previously clicked, making him more sensitive to documents of lower grade.

3.2 Utilities as features

We now turn to the main evaluation of this work: We test whether the Session Utility model estimated *relevance* is useful as a feature to rank documents. We will compare the results of two experiments; We will first use the whole set of features a commercial web search engine uses to rank documents. We then add the *relevance* estimates of the Session Utility model to the set of features and recompute a new ranking function. If the *relevance* estimates is a valuable information, then the ranking using it among its features should outperform the other.

When learning to rank, each query document pair is represented by a feature vector. The vector contains a list of feature values that are deemed useful in computing the document relevance to a given query. Features are divided into query features, document features and query document features. A query feature depends only on the query itself; It is used to characterize queries in different aspects like, for example, the topics or intents of a query. A document feature is only dependent on the document itself and describes document general characteristics regardless of queries, such as PageRank, document class, or spam score of a document. A query document feature depends on both query and document, it describes the level of matching between the query and the document. For a learning to rank algorithm, the performance of the resulting ranking function essentially depends on training data construction and feature design. Training data construction is concerned with how to choose training samples and how to label them. Feature design is concerned with identifying the features that lead to good performance. Provided we already have a set of features, successfully designing a new feature depends on how much more information is provided by the new feature with respect to the existing one, and how strong can the feature indicate query document relevance.

Companies developing commercial search engines have dedicated team of scientist whose function is to derive new features. This is an on going effort that is more than one decade

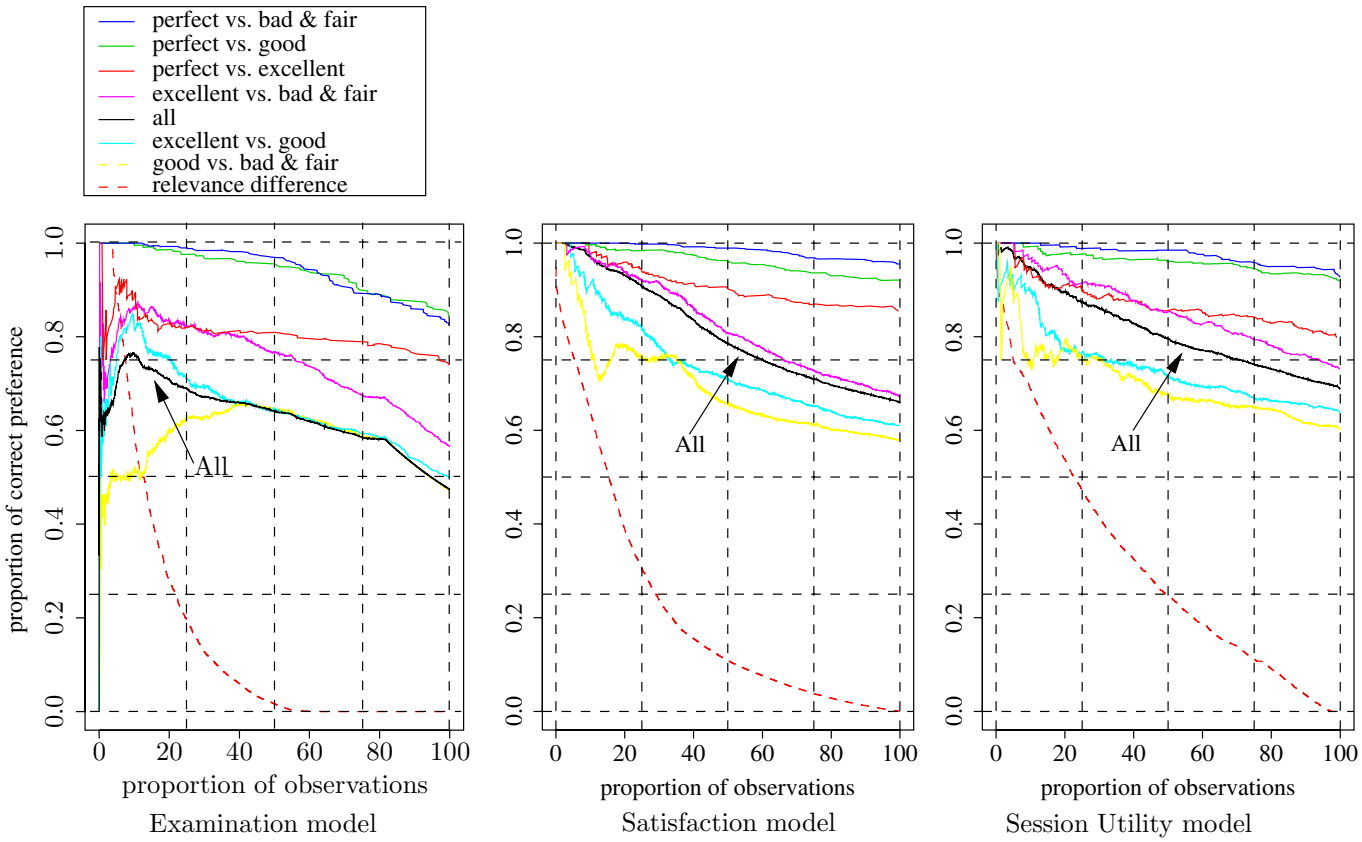

Figure 3: Predicted vs. editorial preferences for the Examination, Satisfaction and Session Utility models. The y-axis is the proportion of predicted document preferences in agreement with the preferences of the editors. The x-axis is the proportion of the total number of pairs used to compute the y-axis proportion. Pairs are ordered by decreasing absolute value of the score difference. The different colored curves represent different classes of preferences according to editorial labels. The three black curves labeled `All` correspond to all the preferences, indistinctly of the class they belong to. The dashed line is the absolute value of the score difference of the observation at the percentile defined by the x-axis. For the reader with a black and white version of this paper, the top-down order in the legend corresponds to the top-down order of the curves in the figures.

long if we consider only search on the web and don't take into account the contribution of traditional Information Retrieval. The point we want to make here is that it is extremely difficult to discover new features that actually manages to improve the DCG. The vast majority of the proposal found in the Literature turn out to be too highly correlated with existing feature to offer any gain. As a consequence, the evaluation we are proposing here is starting from a baseline much higher than is usually found in similar works. In particular, the baseline feature set contains click information like the clickthrough rate, the probability that a document is the last click and in general some version of the features based on clicks found in the Literature.

The metric we use to compare rankings is the the DCG [20]. It is defined as

$$\text{DCG@}K = \sum_{i=1}^{K} \frac{2^{g(i)} - 1}{log(1+i)} \tag{4}$$

where K is the "truncation level" and is set to be $K = 1$ and $k = 5$ in our experiments, and $g(i) \in \{0, 1, 2, 3, 4\}$ is the relevance grade of the i^{th} document in the ranked list. $g(i) = 4$ corresponds to the relevance of a `PERFECT` document, and $g(i) = 0$ corresponds to a document graded as `BAD` by editors. The DCG can be normalized by ideal ranking.

$$\text{NDCG@}K = \frac{1}{Z@K} \sum_{i=1}^{K} \frac{2^{g(i)} - 1}{log(1+i)} \tag{5}$$

where $Z@K$ is the DCG for the ideal ranking, i.e. the ranking obtained by ordering the documents according to their labels.

The Session Utility model requires information about the user sessions, i.e. when a user issues a new query, we need to decide whether the user is still attempting to satisfy the same information need or if he engaged in a different search all together. We used a very simple heuristics to make this decision: If the time between the last action of a user on a page (identified by its cookie) and the new query is below a certain threshold, we consider that the two queries belong to the same session. We then vary the threshold and select the particular value that maximizes the DCG on the training set. More sophisticated technique exist, that involves among other things the similarity between two con-

Table 3: DCG improvement using utilities as feature with two different data sets

data set	DCG@1 gain	DCG@5 gain	NDCG@1 gain	NDCG@5 gain
1	0.42%	0.37%	0.60%	0.15%
2	1.46%	1.07%	1.45%	0.64%

secutive queries or the probability of observing a particular sequence of queries [3]. These could advantageously be used here and are the topic of future work, especially considering that the time threshold technique, no matter how primitive it is, shows some benefit.

In Figure 4, we report the comparison for different shrinkage (a parameter of the Machine Learning algorithm described in [24] and used in this work) and time threshold values for data set #1. It is important to note that including the new feature lead to a DCG gain regardless of the (reasonable) particular set of modeling parameters like shrinkage or time threshold. This adds some extra confidence to the claim that the improvement is not fortuitous. The maximum DCG@5 gain is achieved when the time threshold value is 7 minutes.

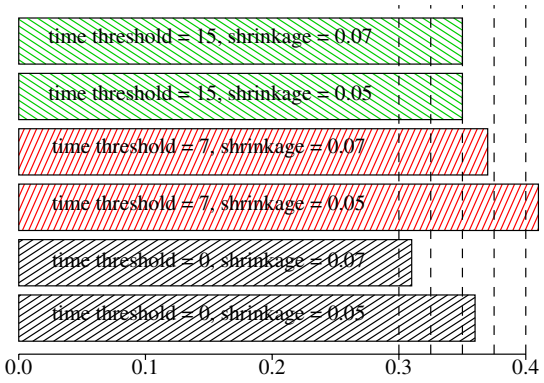

Figure 4: DCG@5 gain (in percentage) using utilities as feature for different values of the time threshold and shrinkage parameters.

In Table 3, we report DCG@1,5 and NDCG@1,5 improvement on the baseline for two different markets where the search engine is leader or on par with the leader in terms of DCG. In each market, the data is divided into training and hold-out sets. Each hold-out set is used for testing and excluded from the training phase. It contains about 5000 different queries in each market. All improvements reported here are statistically significant at 0.05 according to the Wilcoxon *p-value*. We observe a 0.37% DCG@5 improvement in data set #1, and 1.05% DCG@5 improvement in data set #2. DCG@1 improvement is larger than DCG@5.

The Machine Learning algorithm described in [24] permits to rank features according to the importance they have in defining the final document score. We observe that the *relevance* ranks on both market very close from the top: Up to second on one market and around 7^{th} on the other, depending on the run. This is another indication that it provides an information both very useful and uncorrelated with the other features.

We claimed that the Session Utility model is particularly suited to informational queries. To test this hypothesis, we use the fact that editors have classified the queries into navigational and non-navigational queries. We can therefore compare the improvement in DCG@1 and DCG@5 for the two classes. The results are reported in Table 4. We see as expected that the *relevance* as a feature has a significantly larger impact on non-navigational queries than on navigational queries.

Table 4: DCG improvement by query type. The proportion of navigational vs. non-navigational queries is based on the number of distinct queries and doesn't reflect the actual traffic.

query class	DCG@1	DCG@5
navigational (25%)	0.06%	0.28%
non-navigational (75%)	0.76%	0.43%

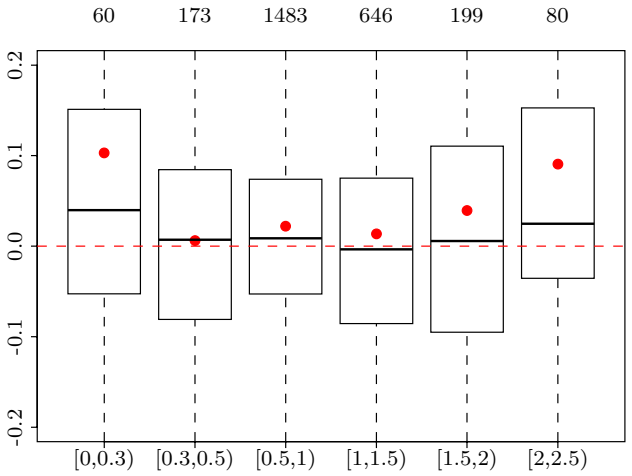

Figure 5: relative DCG@5 gain using *relevance* as feature on the y-axis for different query Click-through rate. Queries have been separated according to their CTR in 6 different bins and the relative DCG@5 improvement for each of the bins is summarized using a box-and-whisker plot. The horizontal segments inside the boxes represent the median improvement for the class. We also added the mean improvement as a filled circle and the number of queries per class above the box-plots.

We explore further the potential of the *relevance* feature on Fig. 5. We computed for each query its average click-through rate CTR), i.e. the number of clicks it receives on any of the documents in the result page, divided by the number of times the query was issued. Queries with a large abandonment rate have a low query CTR by definition. Navigational queries tend to have a CTR close to one because by definition, the user navigates to a single document. Queries with a large CTR are arguably more informational in nature: The user needs several documents to fulfill her information

need. It appears clearly on Fig. 5 that the *relevance* feature has a little or no impact on the ranking of navigational queries, but helps significantly queries that are either hard (low CTR) or informational. This is important because most click based features tend to help identify the target of navigational queries exclusively as reflected by this figure: They already captured the navigational targets and the *relevance* feature offers little gain for them.

Fig. 5 shows that queries with many clicks benefit from a significant mean DCG@5 improvement of around 10%. For the "hard" queries where the query CTR is lower than 0.3, the mean DCG improvement also amounts to around 10%, a very significant amount. The Session Utility model makes the assumption that when the user stops the search, he has satisfied his information need. It seems to go against intuition that such a model help improve the ranking of queries with a large abandonment rate. Part of the explanation for this paradox might be that some of the documents in the ranking are actually relevant, but the snippet isn't. This might explain that users tend not to click, but if they click, they are satisfied.

Discussion and Conclusions

Summary.

The current work proposes a novel way of interpreting the clickthrough data recorded by web search engine logs. Most works in the Literature on this topic concentrate in the perceived relevance and the bias induced by the document position in the ranking. The model we propose by-pass all these problems by concentrating on the user behavior *after* she clicks on a document and not on her decision to click. This leads to relevance estimate based on the document usefulness for the user who choose to click the document instead of its perceived relevance. An interesting consequence is that the relevance is uncorrelated with the document clickthrough rate (CTR). In fact, according to this model the number of clicks on a document is related to the confidence in the relevance estimate, rather than the document relevance itself.

The resulting model is easy and fast to learn. It is easily cast into a binary classification problem of relatively small size: The number of features is the number of distinct documents that have been clicked during the sessions of a query. The number of training examples is the number of clicks during these sessions. Each query is treated separately, making the learning trivial to paralellize.

We used the document relevance predicted by the model as a feature for a "Learning to Rank" algorithm and we compared the results in terms of (n)DCG@1 and (n)DCG@5. Although the feature set of the baseline function contains click features as the CTR and other related statistics, and although the baseline we are comparing against is extremely high –the current ranking function of one of the leading commercial search engine– we observe the new feature leads to a significant improvement in performance. In particular, the feature is particularly useful for queries with a low number of clicks per session in average (less than .3) and for queries with a large average number of clicks (above 1.5). This is particularly interesting because, unlike most click statistics which help with navigational queries, this feature is useful for an harder class of queries: Informational queries and queries with a high abandonment rate.

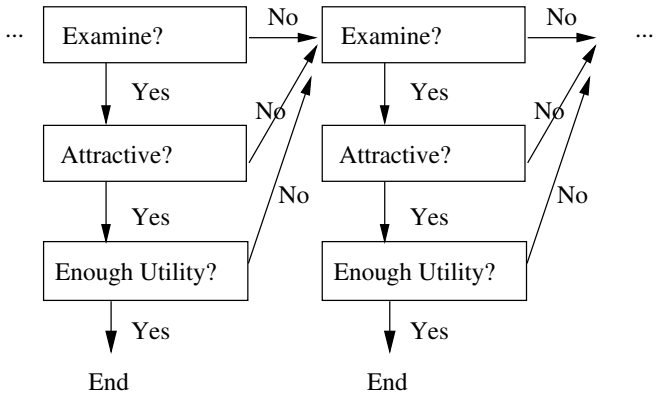

Figure 6: The decision graph of a Session Utility model extended to predict clicks.

Future Work.

In Section 3.1 we compared the new model both in terms of its hypothesis and its performance with two other models proposed recently in the Literature ([10, 6]). It should be relatively straightforward to modify these models and extend them to incorporate the findings of this work. The decision process this suggest is illustrated in Fig. 6. Essentially, a user would first decide whether to examine a position. If he decides to do so, then he would click on the document url if the snippet is attractive. He would then consult the document, a process that update the amount of utility he has gathered. Based on this utility, he would decide to continue the search or not. Such a model has the advantage of being able to predict clicks, and to incorporate the notion that the user accumulates utility until he stops his search.

4. REFERENCES

[1] E. Agichtein, E. Brill, and S. Dumais. Improving web search ranking by incorporating user behavior information. In *Proceedings of ACM SIGIR 2006*, pages 19–26, New York, NY, USA, 2006. ACM Press.

[2] E. Agichtein, E. Brill, S. Dumais, and R. Ragno. Learning user interaction models for predicting web search result preferences. In *Proceedings of ACM SIGIR 2006*, pages 3–10, New York, NY, USA, 2006. ACM Press.

[3] P. Boldi, F. Bonchi, C. Castillo, D. Donato, A. Gionis, and S. Vigna. The query-flow graph: model and applications. In *CIKM '08: Proceeding of the 17th ACM conference on Information and knowledge mining*, pages 609–618, New York, NY, USA, 2008. ACM.

[4] A. Broder. A taxonomy of web search. *SIGIR Forum*, 36(2):3–10, 2002.

[5] 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.

[6] 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.

[7] N. Craswell, O. Zoeter, M. Taylor, and B. Ramsey. An experimental comparison of click position-bias models. In *First ACM International Conference on Web Search and Data Mining WSDM 2008*, 2008.

[8] D. Downey, S. T. Dumais, and E. Horvitz. Models of searching and browsing: Languages, studies, and application. In *IJCAI*, pages 2740–2747, 2007.

[9] G. Dupret, V. Murdock, and B. Piwowarski. Web search engine evaluation using clickthrough data and a user model. In *WWW2007 workshop Query Log Analysis: Social and Technological Challenges*, 2007.

[10] G. Dupret and B. Piwowarski. A user browsing model to predict search engine click data from past observations. In A. Press, editor, *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, 2008.

[11] L. Granka, T. Joachims, and G. Gay. Eye-tracking analysis of user behavior in www search. In *Proceedings of ACM SIGIR 2004*, New York, NY, USA, 2004. ACM Press.

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

[13] F. Guo, C. Liu, and Y. M. Wang. Efficient multiple-click models in web search. In R. A. Baeza-Yates, P. Boldi, B. A. Ribeiro-Neto, and B. B. Cambazoglu, editors, *WSDM*, pages 124–131. ACM, 2009.

[14] A. Hassan, R. Jones, and K. Klinkner. Beyond dcg: User behavior as a predictor of a successful search. 2009.

[15] 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.

[16] T. Joachims. Evaluating search engines using clickthrough data. Department of Computer Science, Cornell University, 2002.

[17] T. Joachims. Optimizing search engines using clickthrough data. In *KDD '02: Proceedings of the eighth ACM SIGKDD*, pages 133–142, New York, NY, USA, 2002. ACM Press.

[18] 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.

[19] T. Joachims, L. Granka, B. Pan, H. Hembrooke, F. Radlinski, and G. Gay. Evaluating the accuracy of implicit feedback from clicks and query reformulations in web search. *ACM Transactions on Information Systems (TOIS)*, 25(2), 2007.

[20] K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446, 2002.

[21] D. Kelly. Implicit feedback: Using behavior to infer relevance. In A. Spink and C. Cole, editors, *New Directions in Cognitive Information Retrieval*, pages 169 – 186. Springer Publishing, Netherlands, 2005.

[22] B. Piwowarski, G. Dupret, and R. Jones. Mining user web search activity with layered bayesian networks or how to capture a click in its context. In *WSDM '09: Proceedings of the Second ACM International Conference on Web Search and Data Mining*, pages 162–171, New York, NY, USA, 2009. ACM.

[23] R. W. White and S. M. Drucker. Investigating behavioral variability in web search. In *WWW '07*, pages 21–30, New York, NY, USA, 2007. ACM.

[24] 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.