Predictive User Click Models Based on Click-through History

Benjamin Piwowarski
Yahoo! Research Latin America
Santiago, Chile
bpiwowar@yahoo-inc.com

Hugo Zaragoza
Yahoo! Research Barcelona
Barcelona, Spain
hugoz@yahoo-inc.com

ABSTRACT

Web search engines consistently collect information about users interaction with the system: they record the query they issued, the URL of presented and selected documents along with their ranking. This information is very valuable: It is a poll over millions of users on the most various topics and it has been used in many ways to mine users interests and preferences. Query logs have the potential to partially alleviate the search engines from thousand of searches by providing a way to predict answers for a subset of queries and users without knowing the content of a document. Even if the predicted result is at rank one, this analysis might be of interest: If there is enough confidence on a user's click, we might redirect the user directly to the page whose link would be clicked. In this paper, we present three different models for predicting user clicks, ranging from most specific ones (using only past user history for the query) to very general ones (aggregating data over all users for a given query). The former model has a very high precision at low recall values, while the latter can achieve high recalls. We show that it is possible to combine the different models to predict with high accuracy (over 90%) a high subset of query sessions (24% of all the sessions).
Key References: [10, 12]

Categories and Subject Descriptors

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

General Terms

Theory, Algorithms, Experimentation

Keywords

Web Retrieval, Query log analysis, User modelling, re-finding, repeat queries

1. INTRODUCTION

Social search is quickly gaining acceptance and is increasingly seen as a promising way of harnessing the common knowledge of million of users to help each other and search more efficiently. Users are increasingly understood as the driving force of the internet and many initiatives are aimed at empowering them. Besides explicit user provided information, there exists very large sources of implicit user information in the internet logs that record user acts. In particular, search engines keep records of their interaction with users in *click-through logs*, which record some sort of user id (through login or cookies), the queries issued by the user, the results returned by the engine and the resulting user clicks.

User clicks on hyperlinks are a soft source of endorsement, since users tend to click on documents they are interested in. For this reason click-through logs are the source of growing attention in the search community. In this work we will analyse query logs from a novel perspective, concentrating on their click *predicting ability*.

In particular we are interested in predicting which document a user will click on immediately after the issuing a query. We are interested in models that can accurately predict these clicks, but more importantly we are interested in models that can predict the *confidence* of the prediction. This is important because we wish to be conservative: we are happy not to make a prediction, but if we make a prediction we want to be confident that this prediction is correct. One can imagine a number of applications for these models, if sufficient accuracy can be achieved. For example, in a result page we could highlight the page that we believe is the one the user will choose, or directly take the user to that page.

We do not pretend that it is possible to predict with high accuracy the target page of a user for most queries. If a user types "Italian cooking" into a search engine, it is hard to tell if the user wants a restaurant, a book, a history page, the page on cooking the user saw last week and really liked, etc. However, we believe that we can achieve high accuracy for *some* queries. In particular, there are two types of queries that we hope to predict with high accuracy.

The first type of prediction we are interested arises when users re-find information. Users commonly follow known paths when searching [3] for information they already found. More specifically, they will tend to re-issue a query when searching for a document they found thanks to a search engine. Our work was motivated by the research on click-through logs carried out by Teevan *et. al.* [10, 11] that analyses this user behaviour. They studied click-through logs and found that 40% of all queries lead to a click on a result formerly clicked *by the same user* during a previous session. Furthermore, 71% of the queries leading to repeated clicks by a user were identical (same string), and 87% of identical query re-writes led to repeated clicks. Only 14% of the queries leading to a repeated click also led to a click on a new (previously unclicked) document. Furthermore, on average 28% of documents clicked by a user received more than one click by that user that year. On the other hand, only 7% of clicks were clicked by multiple users. With this study, Teevan *et. al.* clearly showed the important role that re-finding plays in web search today. They went on to build models which predict whether a user query is a re-find or not.

In this paper, we go one step forward and we build models which predict, not only if a re-find is taking place, but also what is the user's target page, and which is the confidence of our prediction. We refer to such type of models as *user-centric* or *user* models. Furthermore, we generalise the concept of *re-finding* to take into account not only queries issued by the user in the past, but also queries issued by *similar users*; we refer to these as *user-group models*. Finally, we can extend our models to deal with simple navigational queries that *many other users* have issued in the past and agreed on (i.e. "BBC", "KLM", etc.). We refer to these models as *global*.

We note here the difference between this type of predictive models and *personalisation* models. Personalisation models attempt to produce better result rankings; this is done by building a model of the user, and then biasing the general ranking function with the user model to improve relevance. In our work we are trying to achieve something else: predicting with high confidence the target page of a user involved in a re-finding activity. In fact, personalisation is a complementary problem to this one, and indeed personalisation models could be used simultaneously to the prediction models described here.

This paper contains a number of novel ideas:

- it sets up the task of "click-prediction", defines it formally and describes a number of evaluation measures that can be used to test systems,
- it proposes a new framework for click-prediction using standard Bayesian inference models,
- it proposes three alternative models of different characteristics: user centric, global and group based, as well as a combination method.

Furthermore it evaluates the different models as well as a baseline in real click-through data from a commercial search engine. Our results show that predictive models can be used effectively to greatly improve the search experience on the web.

2. PREDICTING USER CLICKS

In this section we will give our formal definition of the problem of predicting user clicks and present a naïve baseline model. Furthermore we will propose some evaluation measures for this problem. To our knowledge, the problem definition and the measures are novel.

2.1 Notation

Assume that we recorded the queries and clicks of users over a period of time. We call a query session $s := (u, q, t, D)$ the tuple denoting a user u, a query q, a time t, and a sequence of clicked documents $D := \left(d_1, ..., d_{|D|}\right)$. For a given session s, we use q_s, u_s, etc. to denote the different elements in s (the query, the user, etc.). We note the sets of users, sessions and queries $\mathcal{U}$, $\mathcal{S}$ and$\mathcal{Q}$. We note $\mathcal{S}_u$ the set of all the sessions by the user u, and $\mathcal{S}_{uq}$the subset of these with the same query q. We can count the different clicks issued by users on specific documents and queries. The basic counts from which the rest are derived are called *click counts*:

$$v_{uqd} := |\{d \in D_s | \, s \in \mathcal{S}_{uq}\}| \, .$$

To simplify notation, we have dropped the dependency of the different quantities with respect to the time t. It will be implicit for the remaining of the paper that any computation or prediction made at time t can made using of (but only of) the sessions in the past $\{s|t_s < t\}$.

Furthermore, we will compute click counts in two alternative ways. We will either count *all clicks* or *single clicks.* All clicks are straightforward counts over the click-through data. When counting single clicks, we eliminate all the sessions for which the user clicked more than once (i.e. $|D| > 1$). Clicking on a single document may be a stronger indication of user satisfaction than clicking on several. If the user did not try anything else, she was probably satisfied with the first answer. On the other hand it is possible that after observing a single document she abandoned altogether or she re-issued a new query.

Our objective is to define models which makes predictions of clicks. We will study this from the point of a Bernoulli process: a sequence of binary events. After a user u issues a query q, we assume that she is presented with a single document d. This event, noted ψ_{uqd}, has two possible outcomes: the user either clicks on the document ($\psi_{uqd} = 1$) or not ($\psi_{uqd} = 0$). This process is repeated many times. We will build different estimators of $\Pr(\psi_{uqd} = 1)$ using different assumptions.

A *prediction* $\Psi_{uq} \in \{W, \emptyset\}$ is a choice of a document for a given user and query at a given time. We want to allow ourselves to be conservative so we allow null predictions ($\emptyset$), particularly for the cases were there is not enough confidence in any particular prediction. A *prediction model* will always follow the two steps:

$$d^*_{uq} = \arg\max_d \left\{\Pr(\psi_{uqd} = 1)\right\} ,$$

$$\Psi_{uq} = \left\{ \begin{array}{cc} d^*_{uq} & if \ \chi_{qud} \geq \rho \\ \emptyset & otherwise \end{array} \right. .$$

where χ_{qud} is some measure of confidence on our prediction d^*. Different estimators of $\Pr(\psi_{uqd} = 1)$ and of χ will lead to different predictions.

There are various ways to deal with equality, i.e. when there is more than one document with same value of $\Pr(\psi_{uqd} = 1)$. One could select one document randomly, or to make a null prediction since there is no preference and we are targeting one click sessions. We chose the latter in our experiments since introducing randomness in our model did not match one of our requirements, controlling the confidence of the model.

2.2 Measuring performances

To measure performances, we define a precision-recall metric. In order to measure different aspects of our prediction, we compute the expectation over the users and queries. Doing so allows us to give, for example, an equal importance to each user or, in the opposite, an importance proportional to the number of times the user asked a query. We consider prediction to be correct if the predicted target page for session s (noted Ψ_s) was clicked by the user in that session (that is, if $\Psi_s \in D_s$).

$$\begin{aligned} \text{recall} &= \mathbb{E}\left(\frac{|\text{predicted sessions}|}{|\text{sessions}|}\right) \\ &= \sum_u \sum_{q \in Q_u} \frac{|\{s \in \mathcal{S}_{uq} | \Psi_s \neq \emptyset\}|}{|\mathcal{S}_{uq}|} \Pr(q|u) \Pr(u) \end{aligned}$$

$$\begin{aligned} \text{precision} &= \mathbb{E}(PM) \\ &= \sum_u \sum_q PM(u,q) \Pr(q|u) \Pr(u) \end{aligned}$$

where:

- Ψ_s is the prediction we made for session s.
- $PM(u,q)$ is the percentage of good predictions for a given query and a given user.
- $\Pr(u)$ is the probability of a user, which can be set to $\frac{1}{|\mathcal{U}|}$ where $|\mathcal{U}|$ is the number of users (uniform probability, equal importance to every user) or $\frac{|\mathcal{S}_u|}{|\mathcal{S}|}$ (importance proportional to the number of sessions)
- Likewise, $\Pr(q|u)$ can be $\frac{|\mathcal{S}_{uq}|}{|\mathcal{S}_u|}$ (importance relative to query frequency) for a query the user asked, and 0 otherwise. Other probabilities like $\frac{1}{|\mathcal{Q}|}$ (uniform probability) are not adapted since the performance measure is not defined for queries the user did not ask.

We experimented with many of these combinations, but it did not change the relative benefit of the models. Therefore we report on the simplest: each session has the same weight:

$$\Pr(u) \times \Pr(q|u) = \frac{|\mathcal{S}_u|}{|\mathcal{S}|} \times \frac{|\mathcal{S}_{uq}|}{|\mathcal{S}_u|} = \frac{|\mathcal{S}_{uq}|}{|\mathcal{S}|}$$

2.3 Naive models

Figure 1a shows the probability that a user will click on a document after a query as a function of the number of times the user has already clicked *on the same document for the same query* in the past. This is equivalent to a model where $\Pr(\psi_{uqd} = 1) \propto v_{qud}$. If we count all clicks, we see that this probability is more than half for documents that have been clicked at least once, and grows quickly to over 0.95. The behaviour is very similarly for *single clicks*, but the starting point is much higher (0.8 instead of 0.55). This confirms the hypothesis that single clicks are more informative. Figure 1b shows the histogram of the number of previous clicks on that document for every user click.

The statistics in 1 give us an idea for a baseline model: predict the most clicked page if it has been clicked above a certain number of times. This can be done with:

$$\Pr_B(\psi_{uqd} = 1) = \frac{v_{qud}}{\sum_{d'} v_{qud'}}$$

$$\chi_{qud} = v_{qud}$$

This will be very precise for queries that the user repeats often to re-find pages. It will be very unprecise for queries that the user issues rarely, or queries for which the user is exploring for new information rather than re-finding. Furthermore, it will not be defined for any queries that the user has not previously typed. We refer to this type of model as a *user-centric* because it depends on the document, the query and the user.

In fact the performance of this estimator can be estimated directly from Figure 1. For example, if we set the threshold to $\rho = 3$, we would make a prediction for roughly 14% of the queries a user types, and we would be correct roughly 95% of the times we make a prediction.

As we can see, this baseline can reach very high accuracy, but this comes at the price of being very conservative (i.e. not making any predictions for 86% of the queries).

2.4 Probabilistic models

We wish to improve over the baseline models proposed above in a number of ways. First, we do not want to have to choose *a priori* a value for the threshold ρ: this value should depend on the click distribution of each query. Very consistent queries should require little evidence, whereas noisy queries should require more. To take this into account we will use Bayesian estimators, which integrate naturally the notion of confidence in the absence of the infinite data. Second, we wish to extend our predictions to queries that the user never typed; for this, we will extend the notion of re-finding.

We wish to model the likelihood $\mathrm{p}(\psi_{uqd} = 1|\mathcal{L})$ considering our data as coming from a Bernoulli process of clicks and no-clicks, where $\mathcal{L}$ is any available past information that can modify our knowledge on ψ_{uqd}. In the Bayesian framework this likelihood is the result of integrating all the possible Bernoulli models (that is over all possible values of the probability θ of success), weighted by their likelihood given past information $\mathcal{L}$:

$$\Pr(\psi_{uqd}|\mathcal{L}) = \int_\theta \mathrm{p}(\psi_{uqd}|\theta)\, \mathrm{p}(\theta|\mathcal{L}) \tag{1}$$

The advantage of using Bayesian estimators is that they will naturally be pulled towards their prior likelihood $\mathrm{p}(\theta)$ in the absence of sufficient data. Choosing the prior likelihoods appropriately, the estimators will be naturally conservative in the absence of strong evidence[1]. This will allows us to *trust* their estimation directly, not only for prediction but also to determine our confidence. In other words, when using Bayesian estimators we will set:

$$\chi_{uqd} = \Pr(\psi_{uqd} = 1|\mathcal{L})$$

This way we can set ρ to some global fixed value (0.95 for example), and the implications of this will be handled by the estimator automatically for every model.

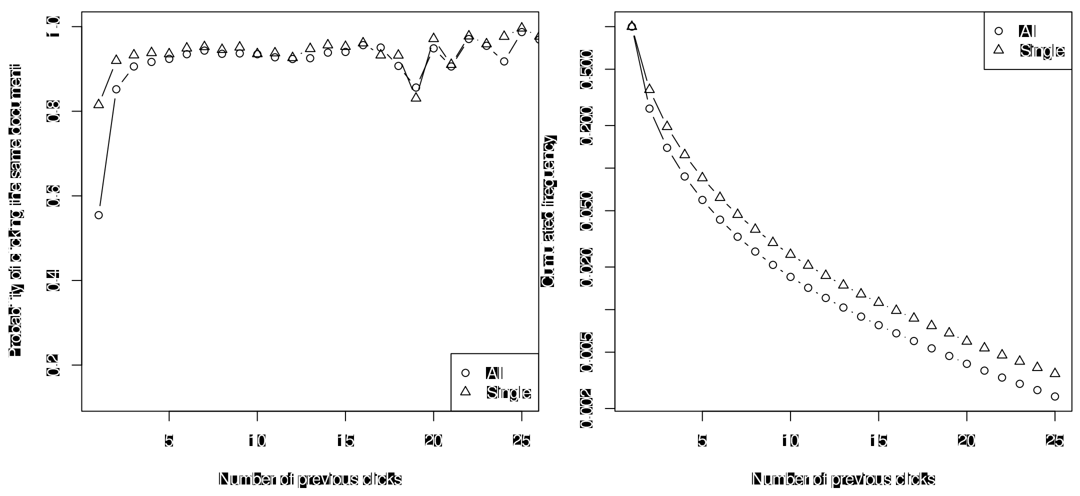
A natural choice of prior for the Bernoulli is its conjugate prior, the beta distribution $\beta(\theta; a, b)$ with parameters $a, b > 0$. These parameters correspond to *pseudo-counts* or fictitious a priori observations of events, in our case clicks on the document or on other documents respectively. Their ratio $\frac{a}{a+b}$ determines the *a priori* likelihood of a click on the document, and the magnitude $a + b$ determine the strength of the prior over the observations.

Without any information, i.e. when $\mathcal{L}$ is empty, the prior is governed by the a priori parameters a and b, and is given by:

$$\mathrm{p}(\theta|\mathcal{L} = \emptyset) = \mathrm{p}(\theta) = \beta(\theta; a, b)$$

As we gather more and more information, we need to update our prior on θ. As the beta distribution is a conjugate prior of the Bernoulli process, the posterior distribution of the parameters when some data is available is a simple count of positive (clicks) and negative (non clicks) events in the Bernoulli process that we add to the fictitious observations. Using the Bayes law, the posterior distribution of a beta distribution, knowing observations coming from a Bernoulli process, is given by:

[1]We note that in frequentist (non-Bayesian) statistics this can also be achieved by smoothing the estimators of frequency (maximum likelihood estimators) so they are robust to small data sets; we have chosen the Bayesian framework instead because we are more familiar with it.



(a) Probability of a click (b) Cumulated frequencies (log scale)

Figure 1: Probability of clicking on the same document given the number of previous clicks. With "all" we count all the clicks while with "Single" only single clicks are accounted for. The data used for this graph is described in Section 3.1.

$$\mathrm{p}(\theta|\mathcal{L}) = \frac{\Pr(\mathcal{L}|\theta) \times \mathrm{p}(\theta)}{\int_\theta \Pr(\mathcal{L}|\theta) \times \mathrm{p}(\theta)} \quad (2)$$

Eq. (2) can be solved analytically (see [6] for a full development):

$$\mathrm{p}(\theta|\mathcal{L}) = \beta(\theta\,;\, a + |S_d|\,, b + |S\backslash S_d|) \quad (3)$$

where S denotes the set of events considered, S_d denotes the subset of positive events and $S\backslash S_d$ the subset of negative ones. In our case, S are the sessions considered in the model and $S = \{s | d_s = d,\ s \in S\}$ the subset of these which lead to a click on d. Plugging Eq. (2) into Eq. (1), we can write the solution to (1) analytically:

$$\begin{aligned} \Pr(\psi_{uqd} = 1|\mathcal{L}) &= \int_\theta \theta\beta(\theta\,;\, a + |S_d|\,, b + |S\backslash S_d|) \quad (4) \\ &= \frac{a + |S_d|}{a + b + |S|} \quad (5) \end{aligned}$$

There are several choices to construct such sets. We will explore two options for S and S_d, namely user-query models and global-query models. In the first case we build a model for each query q of each user u independently, using only data coming for that user, whereas in the second model we mix all the users together into one group:

user-centric model: $\Pr_u(\psi_{uqd} = 1)$ is obtained from (4) using $S = S_{qu} := \{s \,|\, u_s = u,\ q_s = q\}$. We compared this approach to the maximum likelihood (MaxLk) approach, which is simply obtained setting the a priori parameters a and b to 0.

global model $\Pr_g(\psi_{uqd} = 1)$ is obtained from (4) using $S = S_q := \{s \,|\, q_s = q\}$.

In order to fix the values of the hyperparameters a and b, we used a standard optimisation technique [7] described in Appendix A. We found values for a and b of 1 and 0.3 for the user-centric model, and 29.7 and 6.8 for the global one. In both cases, there is a favourable a priori on the document. This was expected since the parameters are computed for documents in the set of those already clicked by the user(s). Note also that the a priori parameters for the global model are much higher than those of the user model. This implies that the user centric model will be much more influenced by new observations.

2.4.1 Dynamic sets of users

In the previous section, we explored how we can confidently predict a document for a user-query pair. We can use this information to build groups of related users. When a user-query pair does not have enough related past information to predict confidently a document, this information can still be used to find an already existing group of users that is close to the user. We construct such groups such that to each group corresponds one and only one document to be predicted with high confidence.

To build such groups, we would like to group users that would click on the same document for the same query. Since this information is not available, we use the user model defined in the previous section, and group users that would be, for a given query, *predicted* the same document. Let us define the *user group* G_{qd} as the set of users that would be predicted the document d for the query q:

$$G_{qd} = \{\, u \,|\, \Psi_{uq} = d\}$$

This definition is deterministic at each time t since predictions are deterministic. We can define click counts for a group as follows:

$$v_{gqd} := \sum_{u \in G_{qd}} v_{uqd} \quad (6)$$

These clicks take only into account the users that are highly related to a group of users who click on d for query q with a high confidence. In other words, it ignores clicks from users who do not prefer the document d clicked. With these counts we can compute our third type of predictive model:

group model $P_{gqd^*}(\psi_{uqd} = 1)$ is obtained by multiplying the probability $\Pr(G_{qd^*} \,|\, u)$ that the user belongs to the group with (4) using the same S_q as for the global model but using the counts from (6).

Computing the probability $P(G_{qd} \mid u)$ can be done in a number of ways. Here we assume that the user belongs to the group unless there is a very low probability that he will not click on the document d, or equivalently if there is some support to the fact that he will click on document d link. The probability that the user belongs to the group is defined formally as:

$$\Pr(G_{qd} \mid u) = \Pr\left[\beta\left(\theta\,;\, a + |S_d|\,,\, b + |S \setminus S_d|\right) \geq \rho'\right]$$

The threshold was set empirically to the value 0.9 using the development set.

The interest of group models is that they *pool* clicks from similar users, allowing the confidence to grow much faster. If a user clicks on a document once, but there are many similar users who also clicked on it, we can be confident that this document is a target.

2.4.2 Combining predictions

Each one of the three models proposed has different characteristics. For example, global models can be very accurate for popular navigational queries whereas user models can be very accurate for user specific re-find queries. Since the models seem complementary, it makes sense to combine them into a single model.

We are currently working on probabilistic models that can combine all the methods, but this work is not completed. Instead, we propose here a simple ad-hoc method to learn the combination of the models. The method is based on one RankBoost algorithm [5] described in Appendix B. We call this model the **RankBoost** model. Other models for combination can be used, but RankBoost had the advantage of working well with ranking models like ours.

3. EXPERIMENTS

In this section, we report experiments with the three models: user-centric, community and global. We report results of individual models as well as the combination model.

3.1 Data

We made use of click-through logs from the Yahoo search engine over a period of 57 days. In order to associate users with a queries, we eliminated from the log all the users for which we did not have a unique ID (i.e. users not logged into the search engine).

In order to establish user sessions we used a timeout of thirty minutes: every click related to the same query and user within thirty minutes was considered to be within the same session. This threshold was suggested by [11]. We verified this threshold with our data: in Figure 2 we plot the distributions of time differences between two consecutive clicks of the same user for the same query (for all users and clicks in the 57 days). We see that the clicks decrease exponentially between 10 and 100. Average query length computed by removing non alphanumeric characters and one letter words was of 2.9 words which is also comparable to values reported in the literature [12, 11].

Queries were not normalised lexically or semantically in any way. This could be an issue, since a small scale experiments about refinding has shown that within a single hour, only 72% of users remembered exactly a query. This is something we may try to improve in future experiments. However it is difficult to normalise queries semantically with introducing much noise (i.e. by confusing two different user intents). A consequence of this choice is that the model favours high precision to high recalls.

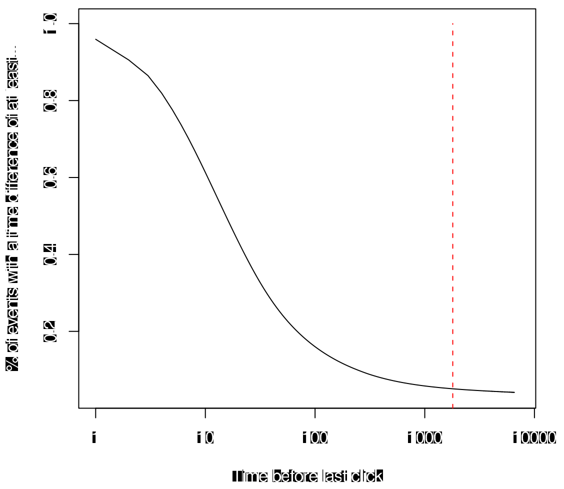
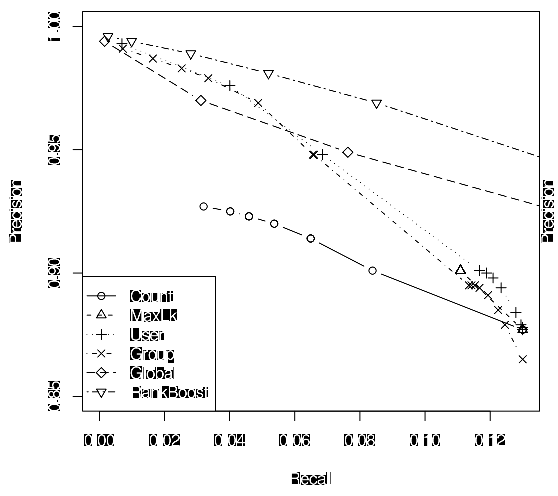


Figure 2: Percentage of events within a given time frame. The 30 minutes limit is the dashed straight line.

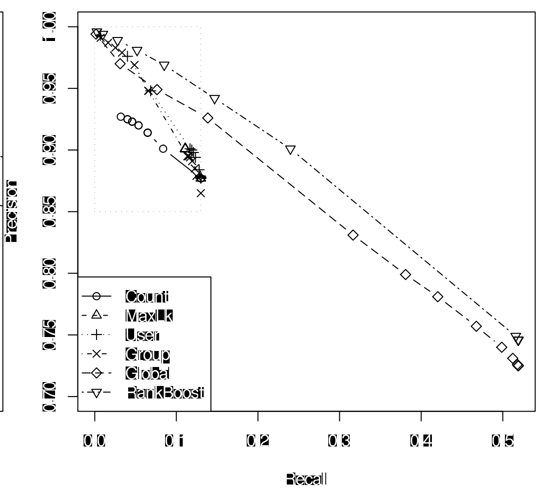
Another open issue is the fact that our query logs did not contain information about URLs that were *not* selected by users, and consequently of the rank of the URLs clicked. This has a direct effect on performance analysis, because we cannot know whether a URL was not clicked upon because the user saw refused it or because it was not even presented to the user. Moreover, as stated in [11], the simple change in presentation order (27% of reclicks are at a different rank as they were for the first click) decrease the number of re-clicks. We ignore these facts in our work.

3.2 Results

In Figure 3, we plotted the precision of the different models with respect to the recall measure described before for *all click* counts. The curves were obtained by using different threshold values on χ_{qud}. The left hand side graph (a) is a zoom of the right hand side graph (b) for low values of recall. In this graph we can see the performance of the user-centric methods, which cannot predict beyond 13% of recall. First we note that the baseline method (Count) performs worse than the other methods in general, but nevertheless it has some predictive power: for 1% of sessions they can predict the target page with 90% accuracy! The maximum likelihood (MaxLk) method performs similarly to the count method, but is not able to distinguish the cases where there was only a few – one or two – sessions from the cases where there are more. Consequently, this model is not able to reach high precision values for lower recall values. Probabilistic models (User, Group and Global) are much better than the baseline for very low recall regions: they can build confidence faster than the counting method. However they are similar to the baseline for high recall levels (beyond 10%). It seems that our naive method of assigning group membership is not sufficient to increase the precision. We plan to investigate this further in the future. Global methods reach much higher recall values (up to 50%) as expected, but surprisingly they seem almost as accurate at the low recall (high rejection) end. The probabilistic global model outperforms the baseline for all values of recall. It also outperforms the probabilistic user-centric models for all recalls beyond 0.06. This is impressive and unexpected. Finally, the RankBoost method successfully combines the different methods producing model that outperforms all others for all ranges of recall. Furthermore, the combination model is almost 5% better than the global one consistently for all values of recall.



(a) low recall range (zoomed view of graph on the right)



(b) all achievable recalls

Figure 3: Recall-precision for different models (the right plot is a blow up of the top left corner of the left plot). Different recalls correspond to different ρ thresholds. For example, if we predict for only the 10% most confident sessions, the combination method achieves over 95% accuracy whereas the baseline count model is under 90%. User-centric methods cannot predict beyond 13% of recall, whereas global methods reach 50%.

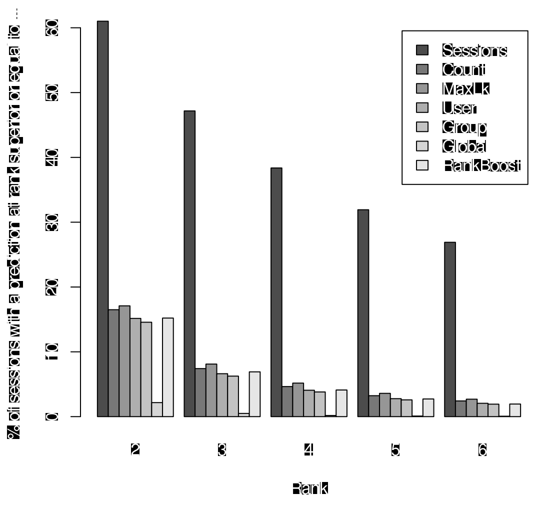


Figure 4: Number of sessions (for predicted or not predicted results) with respect to the rank.

These are very encouraging results. It means that we can hope to predict the target page of a user 50% of the time with an accuracy of 75%, or alternatively 5% of the time with an accuracy of 98%. Such high accuracy can open up the way to new forms of search interfaces which will pro-actively highlight or open the predicted target pages.

In the following we will analyse further two aspects of our models to gain insights on how this accuracy is being achieved by the different models. The first question we pose is: how many of these predictions were already ranked 1 by the search engine without any history information? To some degree, predicting correctly a click on rank 6 is more useful to the user than predicting correctly a click one at rank 1. In Figure 4, we plotted the percentage of predictions we made and the percentage of clicks given a rank. For instance, 9% of the good predictions where documents at a rank superior or equal to 2. This should be compared with the click behaviour of users, who click at rank 2 or more 60% of the time. We can also remark that the more specific strategy, the higher the ranks of the predicted documents, and therefore the more useful the prediction. We note however than even if all our predictions were at rank=1, our models would still be very useful because they do not only pick a document, they also decide *when* to predict and when not to. This confidence feature is useful to design search interfaces that do not solely rank documents; for example, if we wish to highlight the predicted document, open it directly on a browser window, etc.

The second analysis is about the relationship between *single click* and *all click* ways of computing the counts. In Figure 5, we plotted the distribution of the number of clicks per session for our different methods. We can observe that all models predict mostly one click queries (more than 80% of the cases). The combination method makes more predictions for multiple click sessions. It shows that combining the informations from the different models can be very advantageous, since it allows to predicts clicks in multiple clicks sessions.

4. DISCUSSION

Recently there has been some work on user refinding behaviour, which can be roughly divided between *qualitative* studies and *quantitative* studies. Qualitative studies aim at analysing how the user is searching some information he/she has already seen. As users have to be recruited for the experiments, these studies are typically conducted on a small number of users and queries for which detailed information is collected, including the navigation behaviour of the user (going back, using the history of the browser, etc.). On the opposite, quantitative studies use only basic information (clicks, query string, time stamp, etc.) collected by the search engine. They aim at providing statistics on how to characterise clicks of users trying to re-find information. As they use server side log data, they don't need to actively involve the user into the experiments, and hence can be conducted with a great amount of users and/or queries. Our work belongs to this latter category.

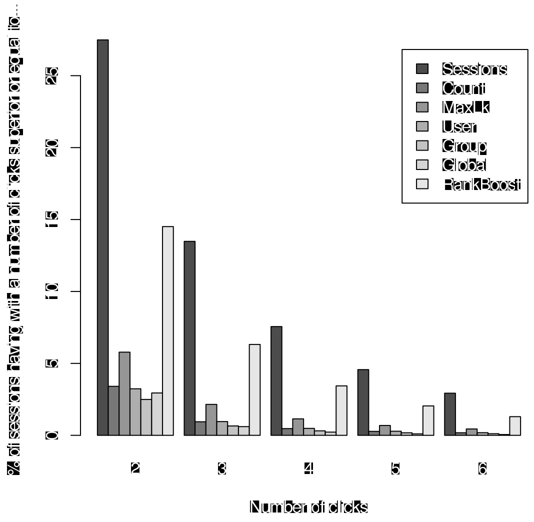


Figure 5: Average number of clicks in sessions in general and predicted sessions.

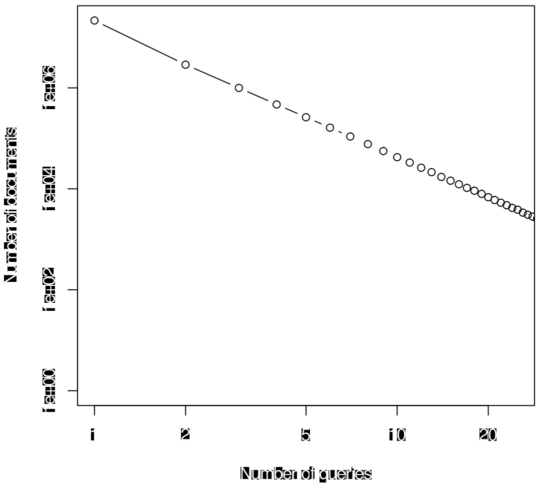


Figure 6: Number of distinct queries per document

Within qualitative studies, we can cite [3, 1], authors study the key strategies for information re-access and underline the fact that search engines are widely used for this task, although this "strategy has problems as it is difficult to remember the exact search terms used to find the material in the first place". While we did not study the query reformulation between queries issued by the same user while searching for the same information, we found out that the already available information is enough to ease refinding.

Wedig et al. [12] studied the amount of data that is available for personalisation, and found out that even when users are not logged it is possible to use cookies: In this case, the amount of information is enough to personalise the search results. They also presented some results on the amount of information provided by a click, and could be useful in an extension of our work to predict user interests in general in order to build other groups of users.

Using web logs, Tauscher and Greenberg [8] have found out that people tend to revisit a considerable number of web pages: They found a 58% probability that the next page visited was previously seen. Noticeably, they also found out that users visit a few web pages frequently. This matches our findings, since people tend to use bookmarks queries for the few web sites they visit frequently, hence providing one possible explication for the good results of our predicting models.

Dumais et al. [4] proposed a system for facilitating information re-use that provide meta-information about the searched results. The Re:Search search engine [9] was designed to help people return to information in the dynamic environment of the Web by maintaining consistency in the search results it returns across time through a mechanism of cache of past queries and merge between past clicked results and current results returned by an external search engine.

As mentioned in the introduction, the work of Teevan et al. [11] inspired this paper. In their work, they focussed on the analysis of navigational repeated queries. They also designed a Support Vector Model to predict whether or not a result would be clicked, and whether or not a previously clicked document would be clicked. The difference with our work is that they did not focus on models where misclassification error can be controlled, and did only consider previous clicks from the same user.

Web search queries can be categorised into different categories [2]. In this article, we have shown that it is possible to use past clicks to predict with high accuracy (over 90%) a high subset of query sessions (24% of all the sessions), that shows that we are able to predict most of the navigational queries with high accuracy.

Finally, we can improve our model by taking into account multiple queries. To investigate if this is worthwhile, we plotted in Figure 6 the number of distinct queries per document. We can see that the far majority of documents are clicked for a unique query, but that there is still a great number of documents that were clicked for more than one query. This shows that there is some information that can be exploited there.

We are currently working on using *complete* (including not clicked results displayed to the user) query logs for a larger time frame (one year) for a subset of users. We don't expect our findings to be changed, although it would be interesting to take into account the time in our models.

Another important direction would be to cope with informational queries. The difficulty lies in the variability of the user clicks and of the queries. It would be interesting to look at query chains (reformulations) for this purpose.

5. REFERENCES

[1] Anne Aula, Natalie Jhaveri, and Mika Käki. Information search and re-access strategies of experienced web users. In Allan Ellis and Tatsuya Hagino, editors, *WWW*, pages 583–592. ACM, 2005.

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

[3] Robert G. Capra and Manuel A. Pérez-Quiñones. Using web search engines to find and refind information. *Computer*, 38(10):36–42, 2005.

[4] Susan Dumais, Edward Cutrell, J. Cadiz, Gavin Jancke, Raman Sarin, and Daniel C. Robbins. Stuff i've seen: a system for personal information retrieval and re-use. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 72–79. ACM Press, 2003.

[5] Yoav Freund, Raj Iyer, R. E. Schapire, and Yoram Singer. An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research*, 2003.

[6] David Heckerman. A tutorial on learning with bayesian networks. Technical Report MSR-TR-95-06, Microsoft Corporation, Redmond, WA, USA, November 1996.

[7] Robert B. Schnabel, John E. Koonatz, and Barry E. Weiss. A modular system of algorithms for unconstrained minimization. *ACM Transactions on Mathematical Software*, 11(4):419–440, 1985.

[8] Linda Tauscher and Saul Greenberg. How people revisit web pages: empirical findings and implications for the design of history systems.

International Journal of Human-Computer Studies, 47(1):97–137, 1997.

[9] Jaime Teevan. The Re:Search engine: Helping people return to information on the web. In *Proceedings of the 28th Annual ACM Conference on Research and Development in Information Retrieval*, Seattle, WA, October 2005.

[10] Jaime Teevan, Eytan Adar, Rosie Jones, and Michael Potts. History repeats itself: repeat queries in yahoo's logs. In *SIGIR*, pages 703–704, 2006.

[11] Jaime Teevan, Eytan Adar, Rosie Jones, and Michael Potts. Information re-retrieval: Repeat queries in yahoo's logs. In *SIGIR'07*. ACM, 2007.

[12] Steve Wedig and Omid Madani. A large-scale analysis of query logs for assessing personalization opportunities. In *Proceedings of the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2006.

APPENDIX

A. SETTING THE A PRIORI HYPERPARAMETERS

In order to set the a priori hyperparameters, we used data from the first part of the logs. For some configurations of $n = |\mathcal{S}|$ and $p = |\mathcal{S} \setminus \mathcal{S}_d|$, we can count the number of times $o(n,p)$ the user clicked on the document d link and the number of time $O(n,p)$ this configuration occurred. We can then compare the observed number of clicks $o(n,p)$ to the expected one $\mathbb{E}(\#\,\text{clicked}/\beta(a+n, b+p))$. As we can suppose that the events (clicking on document after having clicked on it n times out of $n+p$ sessions) are independent, the expected number of clicks is simply

$$\begin{aligned}\mathbb{E}(\#\,\text{clicked}/\beta(a+n,b+p)) &= O(n,p) \times \mathbb{E}(\beta(a+n,b+p)) \\ &= O(n,p) \times \frac{n+a}{n+p+a+b}\end{aligned}$$

We then optimised the mean square error between the observation and the expectation to set the a and b parameters:

$$(a,b) = \underset{(a,b)}{\text{argmin}} \sum_{n,p} \left(o(n,p) - O(n,p)\frac{n+a}{n+p+a+b} \right)^2$$

where there is $o(n,p)$ sessions out of $O(n,p)$ where a user, who had already clicked n times out of N sessions on a given document, clicks on it again. We minimised the function using a Newton-type algorithm [7], and found priors of 1 and 0.3 for the user model, and 29.7 and 6.8 for the global one.

We did not choose to optimise the maximum likelihood as we had many cases where either all the users clicked on the document or none of them did, yielding infinite values in the optimisation criteria.

B. COMBINING MODELS: RANKBOOST

We adopted the RankBoost algorithm [5], a boosting algorithm for combining models. RankBoost is a machine learning algorithm that searches an optimal combination of several weak or uncertain classifiers, which are in our cases the different models for URL prediction. Each of the above described models can compute a score for a document d, a query q and a user u. The corresponding scores are the input to the RankBoost algorithm.

Thanks to the fact that the decisions are binary, we used the simplest version of the RankBoost algorithm (RankBoost.B): At each iteration i of the algorithm, a base model $m(i)$ is chosen, along with a threshold score and a weight α_i. This gives rise to a threshold function f_i which equals 1 if $\mathrm{p}_{m(i)}(\psi_{uqd} = 1)$ is above the learnt threshold and 0 otherwise. As user level models cannot compute always a score, a predefined and learnt value q_i is used for f_i if $\mathrm{p}_{m(i)}$ is not defined (because we do not have any past information for a given model).

$$f_i(\mathrm{p}_{m(i)}(\psi_{uqd} = 1)) = \begin{cases} q_i & \text{if } \mathrm{p}_{m(i)} \text{ is not defined} \\ 0 & \text{if } \mathrm{p}_{m(i)} \leq Threshold_i \\ 1 & \text{otherwise} \end{cases}$$

In our experiments, we performed 1000 iterations of the algorithm which are sufficient for stabilisation of the performance over the training set.

The result of learning is a set of step functions f_i, one for each base model. The final probability of an URL d for a query q is given by

$$\mathrm{p}_{\text{rankboost}}(\psi_{uqd} = 1) \propto \sum_i \alpha_i f_i(\mathrm{p}_{m(i)}(\psi_{uqd} = 1))$$

We furthermore required that each function f_i is decreasing with respect to the score v_i, in order to avoid over-fitting as suggested by the authors. The consequence is that $\alpha_i > 0$ for any i and thus that v_{rb} is an increasing function of the v_i: The learnt function can be seen as a combination of monotonic increasing step functions of $\mathrm{p}_m(\psi_{uqd} = 1)$