Relevance Ranking for Predicting Web Search Results

Pedro M. Teixeira
Doctoral Program in Informatics Engineering, Faculdade de Engenharia da Universidade do Porto
pro11007@fe.up.pt

Abstract. In this paper, we focus on how incorporating user behaviour data on a web search engine can significantly improve the ranking accuracy of top results in real web search engine. A large click log dataset and a set of query-url relevance flags previously labelled by juries was used to train and evaluate a classifier, as well as to build a re-ranking alternative based on a cascade click model. Each one of these implicit feedback based ranking methods can improve the precision of a web search ranking algorithms by as much as 17% compared to the its original rankings.

Keywords: Data Mining, Search Engine, Classification, Click Model

1 Introduction

A web search engine is designed to retrieve information to its user through an organized structure, typically an URL list sorted by their relevance. Since the goal of the search engine is to address the user’s information needs, it is important to access the effectiveness of the retrieval method. The quality of the results can be assessed through explicit relevance feedback, using human assessors to judge the relevance of query-url pairs. However, explicit human ratings are expensive and difficult to obtain and the reason a document meets the information needs of a user group might not be so to another one. Disagreements in relevance judgements are often a reflection of diverse intents being realized with the same query.

Recently, implicit relevance feedback has developed into an active area of research, at least in part due to an increase of available resources. Millions of people interact daily with web search engines; large amounts of interaction data are produced every day, providing valuable implicit feedback through the user’s search session attributes such as query text, timestamps, localities, click-or-not flags, etc. Given a query, whether user clicks a url is strongly correlated with the user’s opinions on the url. However, individual users may behave irrationally or maliciously, or may not even be real users; all of this affects the data that can be gathered [1].

The focus of this research was to explore how implicit measures of user interest (e.g. time data, click through data, query region) could be used to develop
a predictive model of binary relevance (relevant or irrelevant) between a query and an url. The click log and relevance labels were provided from a single source, in this case from a russian search engine [9].

As a baseline model, the search engine’s url rankings were evaluated using the jury explicit information; then, the url list was resorted using a predictive click model and evaluated with the same jury information as described in Section 4.1. Finally, the implicit data was used as features to train a classifier in order to sort the url list by its relevance, as described in Section 4.2. The results are presented and discussed in Section 5.

2 Background and Related Work

Ranking search results is a fundamental problem in information retrieval. The most common approaches in the context of the web use both the similarity of the query to the page content, and the overall quality of a page. A state-of-the-art search engine may use hundreds of features to describe a candidate page, employing sophisticated algorithms to rank pages based on these features. Current search engines are commonly tuned on human relevance judgments. Human annotators rate a set of pages for a query according to perceived relevance, creating the gold standard against which different ranking algorithms can be evaluated. Reducing the dependence on explicit human judgments by using implicit relevance feedback has been an active topic of research.

Several research groups have evaluated the relationship between implicit measures and user interest. In these studies, both reading time and explicit ratings of interest were collected. Claypool et al. [2] studied how several implicit measures related to the interests of the user. They developed a custom browser called the Curious Browser to gather data, in a computer lab, about implicit interest indicators and to probe for explicit judgments of Web pages visited. Claypool et al. found that the time spent on a page, the amount of scrolling on a page, and the combination of time and scrolling have a strong positive relationship with explicit interest, while individual scrolling methods and mouse-clicks were not correlated with explicit interest.

Implicit feedback such as click data has been also used in various ways: towards the optimization of search engine ranking functions (e.g. [3] [4] [5]), towards the evaluation of different ranking functions (e.g. [6] [7] [8]). Most of the works above rely on a core method: to learn a click model. Basically, the search engine logs a large number of real-time query sessions, along with the user’s click-or-not flags; this data is regarded as the training data for the click model, which is used for predicting the click through rate (CTR) of future query sessions. However, clicks are biased with respect to presenting order, reputation of sites, user-side configuration (e.g. display resolution, web browser) [4]. The most substantial evidence is given by the eye-tracking experiment carried out by T. Joachims [8], in which it was observed that users tend to click web documents at the top even if the search results are shown in reverse order.
3 Dataset

The dataset used in this research was gathered from the Yandex [9] search engine. Yandex is the leading internet company in Russia, operating the most popular search engine and the most viewed website. The dataset includes anonymous user sessions extracted from the logs, with queries, URL rankings and clicks [10]. For the purpose of training relevance prediction models, it also includes relevance judgements for the ranked URLs. The logs are about two years old, do not contain queries with detected commercial intent and the user data is fully anonymized.

Some characteristics of the dataset are:

- Unique queries: 30,717,251
- Unique urls: 117,093,258
- Sessions: 43,977,859
- Total records in the log: 340,796,067
- Assessed query-region-url triples for the total query set (training + test): 71,930
- Query-region pairs with assessed urls (training + test): 8,410

The dataset is composed of two distinct entries: the user log, and the relevance labels.

User log The user log represents a stream of user actions, each line representing a query or a click.

A query action is represented in a single line, within the format:

SessionID TimePassed TypeOfAction QueryID RegionID ListOfURLs

A click action is represented in a single line, within the format:

SessionID TimePassed TypeOfAction URLID

Each of the previously stated placeholders is defined as follows:

- SessionID is the unique identifier of a query session.
- TimePassed is the time passed since the start of the session with the SessionID in units of time.
- TypeOfAction is the type of the action. It's either a query (Q) or a click (C).
- QueryID is the unique identifier of a query.
- RegionID is the unique identifier of the country the user is querying from.
  There are 4 possible identifiers (integers from 0 to 3).
- URLID is the unique identifier of an URL.
- ListOfURLs is the list of URLIDs ordered from left to right as they were shown to the user from the top to the bottom. Example:

  10989856  0  Q  10384965 2  671723  21839763  3840421  180513
  10989856 103  C  21839763
  10989856 955  Q  1009161 2  197515  197539  11  179526  5859272
  10989856 960  C  197515
Relevance labels The relevance labels consist of labels assigned by Yandex judges to a subset of URLs appearing in the logs. Labels are binary: Relevant (1) and Irrelevant (0). The judgement of relevance was based not only on the text of the query, but also on the region of the user, but not in every case.

A relevance label is represented in a single line, within the format:

**QueryID RegionID URLID RelevanceLabel**

Where each placeholder is defined as follows:

- **QueryID** is the unique identifier of a query.
- **RegionID** is the unique identifier of the country the supposed user is querying from.
- **URLID** is the unique identifier of a URL.
- **RelevanceLabel** is the relevance label (0 or 1).

Example:

1209161 2 5839294 1
1209161 2 1912415 1
1209161 2 1621201 1
1209161 2 1111 0

4 Methodology

The task we are addressing is the automatic ranking of web search results based on relevance, e.g., given a set of web page URLs \( U = \{ u_a, u_b, ..., u_n \} \) we want to order it from the most relevant to the less for a given query \( q_i \) and a region (user's location) \( r_i \). In order to do so, we used a predictive click model to estimate the relevance between an URL and a query and trained a classifier using implicit data as features.

4.1 Using a Click Model for Relevance prediction

Our goal was to model the relationship between clicks and relevance in a way that would allow us to estimate a distribution of relevance from the clicks on an URL. This can be done using a click model. Two different types of the click models are position models [11] and the cascade model [12]. A position model assumes that a click depends on both relevance and examination. Each rank has a certain probability of being examined, which decays by rank and depends only on rank.

A click on the first URL indicates that the URL is examined and considered relevant by the user, treating the individual URLs in a search result page independently. The cascade model assumes that the user views search results from top to bottom and decides whether to click each URL. Once a click is issued, documents below the clicked result are not examined regardless of the position. With the cascade model, each URL \( u_i \) is either clicked with probability \( r_i \) (i.e., probability that the URL is relevant) or skipped with probability \( 1-r_i \). The cascade model assumes
that a user who clicks never comes back, and a user who skips always continues. Let \( E_i, C_i \) be the probabilistic events indicating whether the url \( i (1 \leq i \leq M) \) is examined and clicked respectively. The cascade model makes the following assumptions:

- \( P(E_1) = 1 \)
- \( P(E_{i+1} | E_i = 0) = 0 \)
- \( P(E_{i+1} | E_i = 1, C_i) = 1 - C_i \)
- \( P(C_{i+1} | E_i = 1) = r_{u_i,q} \), where \( u_i \) is the \( i \)th url

The cascade model was chosen to our relevance prediction due to its simplicity of implementation and calculation speed given the large size of the dataset. With the number of estimation views for a given url, the perceived relevance can be calculated by

\[
\text{Perceived Relevance of url } i = \frac{\text{Number of clicks on url } i}{\text{Number of estimated views on url } i} \quad (1)
\]

The term \( \text{Number of estimated views on url } i \) means the number of times an url has been placed above the last clicked url, respecting the cascade click model. It is emphasized in [5] that a click does not necessarily imply the users satisfaction on the content, instead, the user may have been attracted by some misleading abstracts. Therefore, the introduction of \( \text{satisfaction} \) value is given by

\[
\text{Satisfaction of url } i = \frac{\text{Number of last clicks on url } i}{\text{Number of clicks on url } i} \quad (2)
\]

The term \( \text{last clicks on url } i \) means the last url that was clicked on a given query session. To depict the actual relevance, rather than using perceived relevance alone, one can use the following formula

\[
\text{Actual Relevance of url } i = \text{Perceived Relevance} \times \text{Satisfaction} \quad (3)
\]

Once no training phase is required to predict the url relevance using a click model, the full training set and test set were used to compare the click model results against the correspondent relevance labels from the jury.

### 4.2 Using Supervised Learning for Relevance Classification

The accuracy of click model’s \( \text{actual relevance} \) may not directly translate to relevance. We also used a supervised learning approach to tackle this problem. Therefore the major challenge lies in inferring the proper characteristics (features) of a tri-tuple \( \{q_i, r_j, u_k\} \) in terms of expressing web search result relevance. Table 1 depicts the features to be used in a classifier based on [4] and [5]. Three classifiers were used in this experiment: Naive Bayes, C4.5 and Nearest-neighbour Classifier. The Weka toolkit [13] was used for the experiments reported in this paper. The labelled jury dataset was splitted into training and an unseen test set. The unseen test set is used to report accuracy measures.
Table 1: Features for Relevance Classification Training

<table>
<thead>
<tr>
<th>Feature</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Position</td>
<td>Mean position of the URL in current query, region pair</td>
</tr>
<tr>
<td>ClickFrequency</td>
<td>Number of clicks for this query, URL pair</td>
</tr>
<tr>
<td>IsNextClicked</td>
<td>$\sum$ clicks on next position</td>
</tr>
<tr>
<td>isPreviousClicked</td>
<td>$\sum$ clicks on previous position</td>
</tr>
<tr>
<td>TimeOnPage</td>
<td>URL page dwell time</td>
</tr>
<tr>
<td>CumulativeTimeOnPage</td>
<td>Cumulative time for all subsequent urls after search</td>
</tr>
<tr>
<td>DwellTimeDeviation</td>
<td>Deviation from overall average dwell time on url</td>
</tr>
<tr>
<td>CumulativeDeviation</td>
<td>Deviation from average cumulative time on url</td>
</tr>
<tr>
<td>PerceivedRelevance</td>
<td>NumberOfClicks / NumberOfEstimatedViews</td>
</tr>
<tr>
<td>Satisfaction</td>
<td>NumberOfLastClicks / NumberOfClicks</td>
</tr>
<tr>
<td>ActualRelevance</td>
<td>PerceivedRelevance $\times$ Satisfaction</td>
</tr>
</tbody>
</table>

4.3 Evaluation metrics

We evaluated the ranking algorithms over a range of accepted information retrieval metrics used in [3], namely Precision at K ($P(K)$) and Normalized Discounted Cumulative Gain (NDCG). These metrics are described below.

- **Precision at K**: As the most intuitive metric, $P(K)$ reports the fraction of documents ranked in the top $K$ results that are labeled as relevant. In our setting, we require a relevant document to be labelled Good or higher. The position of relevant documents within the top $K$ is irrelevant, and hence this metric measure overall user satisfaction with the top $K$ results.

- **NDCG at K**: NDCG is a retrieval measure devised specifically for web search evaluation [10]. For a given query $q$, the ranked results are examined from the top ranked down, and the NDCG computed as:

  \[
  N_q = M_q \sum_{j=1}^{K} (2^{r(j)} - 1)/\log(1 + j)
  \]  

Where $M_q$ is a normalization constant calculated so that a perfect ordering would obtain NDCG of 1; and each $r(j)$ is an integer relevance label (0=“irrelevant” and 1=“relevant”) of result returned at position $j$. Note that unlabelled documents do not contribute to the sum, but will reduce NDCG for the query pushing down the relevant labeled documents, reducing their contributions. NDCG is well suited to web search evaluation, as it rewards relevant documents in the top ranked results more heavily than those ranked lower.

5 Results and Discussion

We compared our methods over the search engine’s baseline rankings using the performance metrics NDCG and Precision at K. The performance of the click
model was directly evaluated using the full dataset, once no training dataset was required. In order to evaluate the performance of the machine learning classifiers, a stratified sample of 44% of the full dataset was made to build the testing dataset. The test and training datasets are disjoint sets. We then drill down to examine the effects on re-ranking for the attempted queries in more detail, analysing where implicit feedback proved most beneficial. We first experimented with different methods (Perceived Relevance, Satisfaction and Actual Relevance) of re-ranking the search engine baseline outputs. Figures 1(a) and 1(b) report Precision and NDCG for the search engine baseline rankings, as well as for the click model re-ranking strategies results with user feedback. The improvement is consistent across the top 8 results and largest for the top results: NDCG at $K = 2$ for Actual Relevance is 0.624 compared to 0.493 of the original results, and precision at $K = 2$ similarly increases from 0.490 to 0.625.

![Fig. 1: Precision and NDCG at K for Yandex original rankings, rankings sorted by Perceived Relevance, Satisfaction and Actual Relevance.](image)

### Table 2: Evaluation results of the machine learning classifiers

<table>
<thead>
<tr>
<th>Classifier</th>
<th>TP Rate</th>
<th>FP Rate</th>
<th>Precision</th>
<th>Recall</th>
<th>F-Measure</th>
<th>ROC Area</th>
</tr>
</thead>
<tbody>
<tr>
<td>Naive Bayes</td>
<td>0.47</td>
<td>0.435</td>
<td>0.64</td>
<td>0.47</td>
<td>0.336</td>
<td>0.61</td>
</tr>
<tr>
<td>C4.5</td>
<td>0.596</td>
<td>0.417</td>
<td>0.594</td>
<td>0.596</td>
<td>0.595</td>
<td>0.626</td>
</tr>
<tr>
<td>Nearest-neighbour classifier</td>
<td>0.55</td>
<td>0.462</td>
<td>0.549</td>
<td>0.55</td>
<td>0.549</td>
<td>0.544</td>
</tr>
</tbody>
</table>

On Table 2 is presented the detailed accuracy of the classifiers. The Naive Bayes classifier produces a better precision of 0.64 compared with 0.594 and
0.549 of the C4.5 and Nearest-neighbour classifier respectively. Interestingly, using clickthrough alone, while giving significant benefit over the original search engine ranking, is not as effective as considering the full set of features in Table 1 used to train the classifiers. Figure 2(a) and 2(b) reports Precision and NDCG at K for the different classifiers and also for the Actual Relevance from the click model.

![Figure 2: Precision and NDCG at K for Yandex original rankings, rankings sorted by Actual Relevance and machine learning classifiers: Naive Bayes, C4.5 and Nearest-neighbour.](image)

Our experimental results, incorporating implicit feedback, resulted in significant improvements over the original rankings, using both click model and machine learning classifiers. The set of implicit features, such time region indicator, provides advantages over using click data alone as an indicator of interest. Furthermore, incorporating implicit feedback data as features into the learned ranking function is more effective than using implicit feedback for re-ranking. These results are in accordance with the results reported in [3].

### 6 Conclusion

In this paper we explored the utility of incorporating implicit feedback obtained from Yandex search engine to improve its web search ranking. We performed a large-scale evaluation over the click log, using 71,930 manually labelled queries, establishing the utility of incorporating implicit feedback to improve web search relevance. We compared two alternatives of incorporating implicit feedback into the search process, namely re-ranking with implicit feedback and incorporating implicit feedback features directly into the trained ranking function on a machine learning classifier. Our experiments showed significant improvement over methods that do not consider implicit feedback, gaining improvements as high
Our experiments showed that implicit user feedback can further improve web search performance, when incorporated directly with popular content- and link-based features.

References
