Untangling Result List Refinement and Ranking Quality: a Framework for Evaluation and Prediction

Jiyin He
CWI
jiyinhe@acm.org

Marc Bron
Yahoo! Labs London
mbron@yahoo-inc.com

Arjen de Vries
CWI
arjen@acm.org

Leif Azzopardi
University of Glasgow
leif.azzopardi@glasgow.ac.uk

Maarten de Rijke
University of Amsterdam
derijke@uva.nl

ABSTRACT

Traditional batch evaluation metrics assume that user interaction with search results is limited to scanning down a ranked list. However, modern search interfaces come with additional elements supporting result list refinement (RLR) through facets and filters, making user search behavior increasingly dynamic. We develop an evaluation framework that takes a step beyond the interaction assumption of traditional evaluation metrics and allows for batch evaluation of systems with and without RLR elements. In our framework we model user interaction as switching between different sublists. This provides a measure of user effort based on the joint effect of user interaction with RLR elements and result quality.

We validate our framework by conducting a user study and comparing model predictions with real user performance. Our model predictions show significant positive correlation with real user effort. Further, in contrast to traditional evaluation metrics, the predictions using our framework, of when users stand to benefit from RLR elements, reflect findings from our user study.

Finally, we use the framework to investigate under what conditions systems with and without RLR elements are likely to be effective. We simulate varying conditions concerning ranking quality, users, task and interface properties demonstrating a cost-effective way to study whole system performance.

Categories and Subject Descriptors

H.5.2 [**User Interfaces**]: Evaluation/methodology

Keywords

Simulation; Search behavior; Faceted search; Evaluation

1. INTRODUCTION

Many of today's enterprises require a dedicated search system, i.e., a particular optimal configuration of both a *ranking algorithm* and *interface*, to effectively support their specific type of user, task, and collection. To select this optimal configuration an evaluation metric–capturing a particular user behavior and interface combination–together with a fixed set of queries, documents, and relevance judgements, is required to determine the quality of various ranking algorithms [33]. Traditional batch evaluation metrics, however, typically assume that after launching a query, user interaction remains limited to scanning down a ranked result list and stops at some rank k [4]. As interfaces of modern search systems are equipped with additional elements, such as result filters, *and* users become more actively involved in the search process, the *whole system* effectiveness no longer solely depends on the quality of the ranking, but also on how the interface elements function, as well as on *how* users operate these elements [3, 14, 16, 26]. Consequently, batch evaluation metrics for traditional search systems no longer accurately reflect system performance when it comes to the combination of a ranking algorithm and interface.

A key problem, then, is how to choose between systems with varying combinations of interface elements and ranking algorithms. As a first step, in this paper, we present a framework for comparing search systems equipped with a particular class of interface elements, i.e., elements supporting *result list refinement* (RLR). We define RLR search systems as those that provide: (i) a fixed set of filter values that remain visible to the user at all times; and (ii) the filter values operate on a fixed initial result set for a particular query. For example, a system with minimal RLR elements has a single filter value, where elements of increasing complexity are a list of multiple filter values (keywords/entities), and filter values grouped in categories (facets). A relation exists between facets and RLR elements, however, we do not require filter values to be mutually exclusive, exhaustive [34], or orthogonal [13].

We limit the initial framework to search systems with RLR elements for two reasons. The first is pragmatic: without loss of generality, user interactions with these elements can be modeled as switching between a limited number of different subsets of a result list. It allows this work to go beyond the standard user interaction model in batch evaluation, cf. [4], while remaining tractable. The second is methodological: we wish to focus on user interactions with a class of elements that require users only to *recognize* a suitable filter value to refine a result list with. This in contrast to, for example, query (re)formulations that require additional mental effort on the part of the user [29], thereby allowing greater variability in user interactions depending on individual user characteristics.

Our evaluation framework for systems with RLR elements has two parts: (i) an evaluation measure specified by a model that characterizes how users interact with systems supporting RLR and a specification of how these interactions are associated with user effort and gain (Section 3); and (ii) a simulation strategy, i.e., an instantiation of the interaction model parameters (Sections 4 and 5).

SIGIR'15, August 09 - 13, 2015, Santiago, Chile.

ACM 978-1-4503-3621-5/15/08$15.00
DOI: http://dx.doi.org/10.1145/2766462.2767740.

The framework has two immediate applications. (i) *Prediction*: by obtaining estimates of the parameters of the interaction models from usage data, e.g., from online systems or user studies, system performance can be predicted and evaluated off-line. This allows optimization of systems by varying the ranking algorithm and interface elements, for a particular application and user group. (ii) *Simulation*: our framework allows us to perform "*what-if*" analyses, i.e., to investigate system performance under varying conditions by simulating different combinations of ranking quality, interface elements, and type of users. Such simulation results can inform decisions about the type of interface, ranking algorithm, and queries to be used when comparing systems in subsequent user studies.

To demonstrate the efficacy of our evaluation framework we apply it to the comparison of two standard snippet-based search systems, one with RLR elements and one without. Here, our first goal is to examine the accuracy of the framework in predicting user effort when interacting with an RLR system. We instantiate the model parameters of the two systems with user data and then compare the model predictions with real usage data. Specifically, we seek answers to the following questions: (i) does the effort predicted by our framework correlate with user effort on a task with the two systems; and (ii) does comparing simulated system performance allow us to accurately predict when RLR elements are beneficial and when they are not?

Having validated its accuracy, our second goal is to investigate system performance under varying conditions, including different ranking quality, filter properties, as well as user behaviors—and how these factors interplay. In short, we study the question: "when does an RLR-enabled system help to improve user performance?"

2. RELATED WORK

User interaction models are closely related to effectiveness metrics developed for batch-evaluation: users invest effort to operate a system in order to gain relevant information. With some abstraction, a batch evaluation metric can be viewed as a function that makes predictions about user effort and gain based on its user interaction model, i.e., assumptions about how users operate the system, cf. [4]. For example, widely used metrics P@K and NDCG@K employ user interaction models of two different types of user search behavior: P@K assumes users examine all top K results in no particular order, and NDCG@K assumes users examine top K results from top-to-bottom and find lower-ranked results of less value. In both cases, we can interpret the number of documents examined as effort, and the number of relevant documents found as gain.

User interaction with a traditional search interface. Modeling user interaction with retrieved results has become a central topic in recent discussions on evaluation methodology [1]. A wide range of stochastic models have been learned from Web search engine click logs [5, 8, 11, 12]. Common to these models are a few assumptions that have been identified as typical user interaction patterns in Web search. For example, the *examination assumption* states that users are less likely to view lower ranked results [21]; and the *cascade model* assumes users browse a ranked result list from top to bottom, and stop once a relevant result is found [9].

Meanwhile, several effectiveness metrics have been proposed in an effort to integrate more realistic user interaction models. For example, the Rank Biased Precision (RBP) [28] models the "persistence" of a user, i.e., how likely a user examines a next result when going down a ranked list; the Expected Reciprocal Rank (ERR) [6] is derived from the cascade model; and Chuklin et al. [7] proposed to turn click models into evaluation measures. Time-biased gain takes into account user variability in terms of the time needed to process information, e.g., reading summaries [31].

An alternative use of user interaction models is to *simulate* user search activities, in order to evaluate system performance under various conditions such as "what if users do X?" which are not likely to be investigated by a user study. For example, Smucker and Clarke [31] presented a method to simulate time-biased gain. A shared assumption in the studies listed so far is the traditional ranked list search interface.

Studies that go beyond the traditional *single* ranked list based user interaction is session-based evaluation [20, 22]. Here, in addition to the examination assumption, an extra user decision—whether to reformulate their query—is modeled (cf. Section 3.1).

Beyond traditional search interfaces. Moving away from the traditional ranked list search interface, Fuhr [14] proposed the interactive probabilistic ranking principle (iPRP) aimed at providing a formal description of user interactions and the corresponding optimization strategy for a system. However, instantiation of this general model for practical use remains an open problem.

More concretely, faceted search is a typical example where user interaction models need to go beyond the assumptions of a ranked list interaction model. Often, simulation based evaluation is employed [23–25, 27, 32]. A key notion shared by these simulation models is "*utility*" [23, 25, 32]—the trade-off between the effort users spend and the benefit gained, e.g., finding a target document.

Various heuristic user interaction models have been proposed for simulation. These models assume different user goals and how users interact with retrieved results and facets. In terms of user goals, Kashyap et al. [23] assumed users examine all filtered results in the context of database queries. Alternatively, in [25, 30, 32] users were assumed to find only one relevant result. In terms of user operation with facets, Koren et al. [25] assumed that users can always recognize the facet that contains relevant document(s), and select facets in one of the following ways: (i) randomly; (ii) facets with least document coverage; (iii) the first facet that contains the target document; or (iv) the optimal facets. Kong and Allan [24] assumed that users sequentially scan facets and skip an entire facet when they find it irrelevant.

Our work. Our goal is to devise a new evaluation method for systems with a search interface enabling RLR elements. What we need is a user interaction model that is able to characterize not only the traditional "examine a result list" interactions, but also interactions with RLR elements.

We evaluate an RLR system under the same notion of utility (user effort and gain) as in the above studies in faceted search. Our work differs in two important ways. First, we model user interactions with RLR elements (including facets) in a more natural way—users scan filtered results without a particular order; and they may and may not recognize a "good" filter value. Second, we do not make explicit assumptions to create categories of users (like in [25]). The variability of users is captured by a probabilistic framework: by varying two model parameters, we are able to simulate a wide range of users. This second property of our model allows us to encompass empirical user interaction models developed for traditional search interfaces, i.e., to fit the model with real usage data and make predictions of system performance with respect to a particular group of users/search tasks.

3. MODELING RLR INTERACTIONS

Our evaluation framework has the following components: (a) a user interaction model that characterizes how users interact with a system that enables result refinement (Section 3.1); (b) associating

effort and gain to user interactions for evaluation (Section 3.2); and (c) integration of the above two components (Section 3.3).

We assume that users perform *actions* to make progress on a search task (e.g., inspect results); every action costs *effort*; and the user may *gain* from that action by finding relevant information. With an interaction model, we simulate and predict *action paths* of users during a search task, which vary across users and search tasks, and are influenced by the quality of the result lists. By associating effort and gain with different paths of search actions, we are able to predict user effort given different types of users, tasks and the quality of result lists.

Overall, the effectiveness of a system can be measured by answering questions such as: How much effort is required to achieve x amounts of gain with system A, as compared to system B?

3.1 User interaction at a conceptual level

User interaction with a basic interface. With a basic search interface, the common assumptions are: users browse a result list from top to bottom; and after examining each result, they make a decision—whether to continue to examine another result, or to give up this result list [12, 15].

User interaction with an RLR-enabled interface. With an RLR-enabled interface, apart from examining results in the retrieved ranked list, users may choose to refine the result list by filtering on a particular value. A typical consequence of these RLR interactions is that users switch between different filtered versions of the original result list. We refer to these different versions of the result list, including the original ranked list, as *sublists*.

Without making assumptions about the specific implementation of these elements, at a functional level, we can model the user interactions with RLR elements as *selecting a sublist*.

This leads to at least two additional decisions a user needs to make: (1) continue with the *current* sublist, switch to a different sublist, or quit searching? and (2) if switching, *which* sublist to select next?

Parameterization of user interactions. Each of the decision points introduces uncertainty in computing user effort and gain during a search task: it is at these points users diverge from each other's action paths. Taking these decision points as variables of our interaction model allows us to capture variability in user behavior.

Specifically, to quantify the uncertain nature of user decisions, we model the outcome of the aforementioned decisions as random variables following specific distributions:

- **Continuation decision:** we model the decision of user u at rank r to examine the next result in the *same* result list as a binary variable $s_{r,u} \sim Ber(p_{r,u})$. While it looks similar to the *persistence probability* [28], we do not make the i.i.d. assumption about the continuation behavior at each rank. That is, in [28], a single persistence probability p is shared by results at all ranks; and the probability that a user examines the result at rank r is p^{r-1}. The Bernoulli parameter $p_{r,u}$ in our model, however, is specific to a rank r and a user u, thus leaving more flexibility for setting different hypothesized values for simulation or fitting empirical parameters from log data.
- **Switching decision:** similarly, we model the decision of user u at rank r to switch sublists as a binary variable $l_{r,u} \sim Ber(p^l_{r,u})$ with parameter $p^l_{r,u}$ specific to a user and a rank.
- **Sublist selection decision:** we model user decisions on sublist selection among K candidates as a vector of binary random variables following a categorical distribution $\mathbf{f} \sim Cat(K, \mathbf{c}_u)$. Sublist k is chosen if $f_k = 1$; 0 otherwise; and only one list is chosen at a time. The parameter $\mathbf{c}_u$ determines the likelihood that a list is chosen by a user, e.g., according to his/her perception of the quality of the sublists.

These three probabilities can be set to empirical values estimated from usage logs (Section 4), or based on hypotheses about their values (Section 5). Of course, users may decide to quit searching. However, as quitting is complementary to the continuation and switching decisions, there is no need to explicitly define it.

3.2 User actions, effort, and gain

With the conceptual user interaction model in place, we now specify how we can associate effort and gain to user interactions.

User actions. We consider 3 types of action:

- **Examine result:** Users examine a result to determine its relevance, by inspecting the title, summary, or the document content. While these introduce great variety in the effort needed for an examination, we consider them as a *single* action of constant cost. We wish to focus on factors that change the result list, e.g., pagination or filtering, which may lead to more examination activities. However, it is straightforward to incorporate fine-grained levels of user interactions with our model, which we leave as future work.
- **Pagination:** While not explicitly modeled as a decision, a user will need to paginate if he/she decides to examine a result which is on a next page. Effort required by pagination is directly related to result list quality.
- **Select candidate result list:** This activity involves a series of mental activities such as estimating which sublist is more likely to contain relevant information, e.g., by inspecting the filter names. As with result examination, we abstract away the details and treat the whole process as a single action.

Effort of actions. Each action is associated with an amount of effort. Let A be the possible actions users can perform, and $P_a = a_1, ..., a_t$ be the action path of a user that starts browsing results of q until he/she stops, $a_i \in A$. The user's effort along P_a is

$$\mathcal{E}(q, P_a) = \sum_{i=1}^{t} w_i a_i, \quad (1)$$

where w_i is the effort needed for action a_i.

Effort can be implemented in different ways. For example, with NDCG or P@10, the only actions considered are "examine" a document, and each costs a unit effort. With time-biased gain [31], effort is implemented in a more elaborate way, e.g., the effort required for an "examine" may depend on the document length.

Gain of actions. We assume that user gain is determined by the relevant documents they encounter. Let D_{P_a} be the documents a user encounters along action path P_a; its total gain is

$$\mathcal{G}(q, D_{P_a}) = \sum_{d \in D_{P_a}} rel(d, q), \quad (2)$$

where $rel(d, q)$ is the relevance judgement of d w.r.t. the query q. This approach to "measuring" effort and gain is of course closely related to the cumulative gain type of evaluation measures [2, 19].

3.3 User action paths

The final ingredient that ties together the above components for measuring user effort/gain when interacting with an RLR system is the user action path P_a. Assuming users examine documents in a ranked list from top to bottom with a basic interface, the order in which users examine documents is deterministic. The only uncertainty is that users may quit, which can easily be handled by computing the gain at a cut-off point, or at an expected search depth [28]. However, with an RLR interface, the possible paths users can take for a query is combinatory given that users can switch

between sublists without a particular order. Thus we resort to a Monte Carlo method. The interaction model specified in Section 3.1 allows us to simulate possible user action paths for a given set of parameters, creating a sample of user effort and gain for a particular task and user behavior. System performance can then be compared with these samples using standard statistical tools.

Action path constraints. To further reduce the complexity of the simulation process, we constrain the possible user action paths with the following assumptions.

A1 Users examine results in a ranked list from top to bottom.

A2 When switching between sublists, users *skip and only skip* the results they have already seen. This prevents inflated counts of user gain (e.g., in terms of relevant documents encountered). A similar design choice has been made in the literature for evaluating faceted search systems [30]. In addition, when switching, users always examine the next (unseen) result in the new sublist, preventing an infinite switching loop. If all documents in a sublist are exhausted then its filter value is no longer selected.

A3 Instead of assuming a user quits searching with a certain probability, we assume a deterministic cut-off, leaving two complementary decision points with continuation probability being sufficient. Practically, for gain-based measures, the cut-off is based on a fixed amount of effort, and systems are compared in terms of the amount of gain achieved for a fixed amount of effort (cf., NDCG@K). For effort-based measures, a cut-off can be set to gain and systems are compared in terms of the amount of effort needed to achieve a fixed amount of gain (cf., expected search length).

Steps to simulate a user action path.

1. Specify model parameters (with empirical or hypothesized values), i.e., $p_{r,u}$ for continuation decisions, and $\mathbf{c}_u$ for sublist selection decisions.
2. At each step, draw $s_{r,u} \sim Ber(p_{r,u})$. If $s_{r,u} = 1$, add *examine* to P_a; else draw $\mathbf{f} \sim Cat(K, \mathbf{c}_u)$, add *select sublist* and *examine* to P_a. When encountered, add *pagination* and handle situations specified in A2.
3. Stop when: (1) the total effort/gain meets a predefined cut-off value; or (2) all results are exhausted.

4. VALIDATION OF PREDICTION

The core of our framework consists of a specific mapping $\hat{y} = h(\vec{x})$, where $\hat{y}$ is the estimated user effort, aimed at approximating the actual user effort y, given the input variables $\vec{x}$. In a typical machine learning scenario $h(\cdot)$ would be *selected* from a pool of possible hypotheses by fitting example pairs of y and $\vec{x}$. In contrast, we have *specified* in advance a single hypothesis h^*, i.e., the interaction model motivated in Section 3, and the values of $\vec{x}$ is determined by specific types of user behavior. Here, we validate our hypothesis h^* by examining how well its output, $\hat{y}$, approximates actual user effort y.

In this section, we validate our model using usage data gathered for an RLR interface. Specifically, we calibrate our interaction model parameters ($p_{u,r}$ and $\mathbf{c}_u$) with empirical values derived from the usage log of a particular group of users (i.e., participants of our experiment), and examine whether the model prediction corresponds to the actual user effort as recorded in the log. We aim to answer the following questions:

Q1 Does the predicted effort ($\hat{y}$) correlate to user effort (y) as computed with usage data?

Q2 Can we accurately predict when an RLR interface is beneficial, compared to a basic interface?

4.1 Obtaining usage data

4.1.1 Test collection

We use the TREC Federated Search data [10], for it has two important properties: (1) all document-query pairs in this collection have been assessed. This provides us with a more accurate estimation of system performance; and (2) this collection has been constructed by querying a large number of web search engines all of which are categorized. These categories can be converted to filter values in an RLR system. The complete list contains 24 categories [10, Table 3], including academic, travel, etc.

The collection contains 50 (judged) test topics, their associated web pages, and the summaries (snippets) to represent these pages. We create rankings for each topic based on a standard query likelihood model as implemented in Indri.

We consider two types of system: one with a basic interface, and the other with an RLR interface, where categories are used to construct sublists for each topic. We treat every document as being annotated with the category of its source search engine. Since an engine can be in multiple categories, and documents may have been retrieved by multiple engines, every document is associated via its source(s) to one or more categories.

Relevance judgements for the Federated Search track are graded, 4 levels from highly relevant to non-relevant. We only distinguish between relevant (level 1–3) and non-relevant to ensure more than 10 relevant documents per topic are available (see Section 4.1.3).

4.1.2 Search interfaces

RLR interface. Fig. 1 shows a screenshot of the RLR enabled interface, where numbers 1–6 indicate components of the system. On the left are the filter values (1) as provided by the federated search track [10]. On the right a dashboard (2) is available indicating the number of clicks left for a task and the number of relevant documents found. After 25 clicks a "give up" button (3) would appear providing the option to skip the remainder of the task. The topic description (4) is available at the top of the screen. An additional button allows users to expand the description and review the examples as provided before starting a task. The middle of the screen is devoted to a scrollable result list (5) with 10 snippets. At the bottom of the page a pagination button (6) is available. See [17].

Basic interface. The basic interface is similar to this design except that the filter panel (1) is unavailable. One of the filter values in the RLR interface is "All," which is the unfiltered list as it would be in the basic interface. It is possible for users to ignore the filter values and use the RLR interface exactly as the basic interface.

4.1.3 Setup of user experiment

We investigate the *effort* it takes users to locate relevant documents. While our interaction model can be applied to both effort-based and gain-based measures, in this study design we focus on measuring effort.

User task. Users were asked to find 10 relevant documents for a topic. It is not as trivial as, e.g., finding 1 relevant document, allowing variability between user action paths. Meanwhile, it limits the effort required to a manageable amount.

Specifically, we asked users to locate by clicking on 10 result summaries of relevant documents within 50 clicks, where a click is counted if it is on a result summary, a pagination button, or a filter value. We use the 50-click limit to prevent users from clicking every result and to force users to make conscious choices instead. We require users to only click on summaries to abstract away from actions such as opening and reading documents as well as to keep the

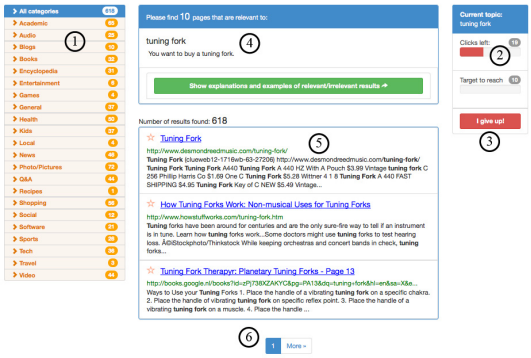


Figure 1: The RLR interface: (1) filter values; (2) dashboard; (3) give up button; (4) topic description; (5) result list; (6) pagination. The basic interface excludes (1).

time necessary to complete a task manageable. To reduce user variability in judgements of relevance, we provide feedback to users whether a clicked result is relevant or not.

Experiment design. We recruited participants via university mailing list and social media. We used a standard between-subject design common to A/B testing, where each new user is randomly assigned to one of the two interfaces and directed to the same interface on subsequent visits. To reduce learning effects, new tasks are randomly assigned to users.

4.1.4 *Obtained usage data*

In total 145 task instances were completed by 49 users for the system with the basic interface, and 255 by 48 users for the RLR interface. The median number of completed task-instances per task is 2 for the basic interface and 3 for the system with an RLR interface. As some tasks have been completed by more participants than others, we consider median values in our analysis.

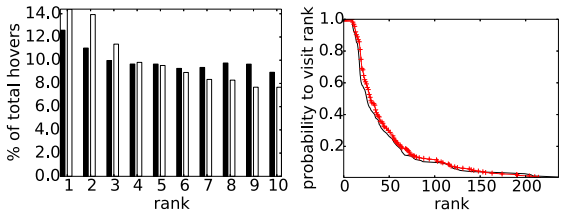
4.2 Measuring user & predicted effort

To answer Q1 and Q2, we need to compute the following quantities: (i) user effort y (with basic or RLR interface) as derived from the usage data, and (ii) the predicted effort $\hat{y}$, where the model parameters are calibrated with the actual usage data.

4.2.1 *User effort*

For simplicity we assume equal effort for user actions. We measure user effort (y) as the number of result summaries users visited and the clicks they spent on choosing filter values and pagination.

To determine which summaries a user has visited we consider mouse hovers over results, which has been shown to correlate with eye-gaze [18]. Fig. 2(a) shows the percentage of total hovers over the 10 ranks of each result page. We observe that the distribution of hovers over the ranks is relatively uniform, i.e., compared to the distribution of hovers over ranks on Web search engine result pages where differences of 38% between the highest and lowest rank are observed [18]. We observe a difference of 3% between the highest and lowest rank for the basic interface (back bar), and 6% for the RLR interface. That the skew is slightly stronger for the RLR interface is expected as not all filter values return 10 results. The hover data suggests that participants tend to visit all result summaries on a page. This is by design as in our systems we reduced the effort needed to judge a result page (i.e., read summary, open/read the page) to judging the result summary. We are therefore able to focus on user effort as introduced by interaction with RLR elements and not by pages and result summaries.



(a) Distribution of hovers over the 10 ranks for basic (black) and RLR (white hatched).

(b) Probability of rank visit for basic (black) and RLR (red hatched) interface.

Figure 2: Users' result page rank visiting behavior.

Given these observations, we approximate the number of result summaries visited as follows. We assume that if a user does pagination, they have seen all the results in the previous page and count all results in SERPs before the last visited page as visited. On the last page, we count the number of results up to the last clicked result.

4.2.2 *Predicted effort*

So far we have obtained user effort y by directly counting the number of actions recorded in the usage log. To compute $\hat{y}$—an approximation of y of our user group—we calibrate our model parameters with empirical values derived from the same log data.

Continuation probability. We compute $p_{r,u}$ using the empirical distributions of the search depths aggregated from all users. The probability that a user will examine the result at rank r is computed as the number of times r has been visited, normalized by the maximum number of times a rank has been visited. In terms of the number of results a user has visited, we take the same approach described above (Section 4.2.1). Fig. 2(b) shows the probability of visiting each rank for the basic and RLR interface. Under the assumption that participants visit all result summaries on a page, ranks 1–10 are visited an equal number of times.

Sublist selection probability. To model user preferences of sublists (filter values), we collect the counts of filter value clicks for each query, and set these as the parameter $\mathbf{c}_u$. That is, the expected value of the probability a filter value will be chosen is proportional to how often it is chosen by the users. Since the original result list, i.e., the filter value "All," is always shown to the users as a starting point, we always add 1 count to it.

4.3 Predicted vs. user effort

We simulated a sample of 1000 $\hat{y}_r$ for an RLR interface. To answer Q1, we perform a correlation analysis between the median of predicted effort ($\hat{y}_r$) and the median of user effort (y_r) over the 50 topics. We see a significant linear correlation between the two (Fig. 3): Pearson $\rho = 0.79$ ($p < 0.001$). This suggests that our proposed model can be used to reliably predict user effort needed in accomplishing a search task in terms of the number of results visited and the number of filter values they need to explore.

4.4 Predicted vs. user benefit

Next, we investigate how the predicted effort can be used to compare system effectiveness, e.g., between a system with a basic interface and with an RLR interface (Q2). To proceed, we investigate whether and on which topics an RLR interface reduces the effort needed to complete the task, as compared to a basic interface.

We compute two quantities for each topic (see Fig. 4):

1. User difference: $\Delta effort_u = y_b - y_r$
2. Predicted difference: $\Delta effort_p = y_b - \hat{y}_r$

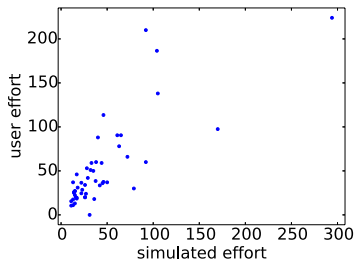


Figure 3: Correlation between estimated effort and user effort.

Note that y_b (effort spent with the basic interface) is fixed as the action path of a user with a basic interface is deterministic.

In terms of user difference, we observe that on 31 out of 50 topics less user effort is needed with the basic interface ($\Delta effort_u < 0$); on 16 topics effort is less with the RLR interface and on 3 there is no difference. Of the 31 topics where $\Delta effort_u < 0$, on 14 a user is able to save more than 10 actions (points of effort), e.g., paginating to a next page and scanning 10 results. Of the 16 topics where $\Delta effort_u > 0$, on 11 a user is able to save more than 10 actions.

On 34 out of 50 topics, $\Delta effort_p$ agrees with $\Delta effort_u$, as in, which interface would reduce user effort in completing a topic. All cases of disagreement are on topics where users of the RLR interface spent more (or the same) amount of effort than users of the basic interface. In these cases participants may have struggled to effectively use filter values or did not use filters at all. If participants did effectively use filters, our interaction model predicted that using filters would save effort.

We observe that our model is able to identify 100% of the cases when an RLR interface is better (i.e., costs less effort) than a basic interface. However, when it predicts that an RLR interface is better, it is only correct in 52% of the cases (0.68 F1-measure). In contrast, it is able to predict when a basic interface is better with 85% precision and 55% recall (0.66 F1-measure). This suggests that if a user would benefit from using the RLR interface, then the model will predict so, and if our model indicates that the basic interface is more beneficial, use of the RLR interface should be avoided.

Further, we find that $\hat{y}_r$ has significant negative correlation with $\Delta effort_u$: $\rho = -0.49$, $p < 0.001$, confirming its predictive power. A negative correlation means that the higher the effort needed with an RLR interface predicted by our model, the more likely that a basic interface is better. The stronger this correlation is, the better the model is at predicting which interface is beneficial.

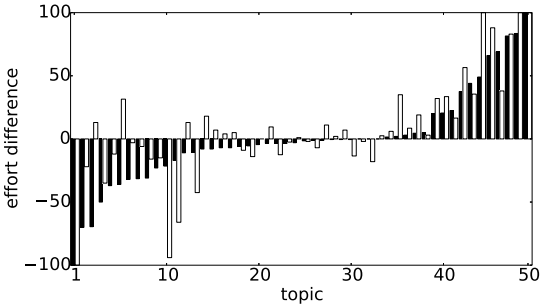


Figure 4: The black bars show $\Delta effort_u$, and the white (hatched) bars show $\Delta effort_p$. A negative value indicates more effort is spent with the RLR interface. Difference values exceeding 100 or less than -100 have been cut-off for legibility.

5. WHOLE SYSTEM PERFORMANCE

In Section 4 we have demonstrated that our user interaction model is simple, tractable, and able to accurately predict user effort with empirical parameter settings. We observed that an RLR interface can be useful for some queries while the basic interface is good for others. Many factors may contribute to this observation, ranging from system properties (backend as well as UI) to user properties. However, it is difficult to investigate the exact impact of these different factors with user studies: a large number of experiments are needed given the number of factors considered and their combination; and it is difficult to control user behavior.

In this section, we discuss how our evaluation framework can be used for studying whole system performance, under strictly controlled and varying conditions that may not be attained in real life studies. We use the same test collection as before, and focus on one question: **Q3.** When does an RLR interface help?

By instantiating our interaction model with parameter values designed to reflect user behavior with desired properties, we generate simulated usage data of large quantity and under strict control. We then study *whether* and *how* various factors (and their interactions) affect the advantage of an RLR interface versus a basic one.

5.1 Simulating user browsing behavior

To start with, we describe how we instantiate the interaction model with parameters that characterize different user behaviors.

Examination depth on a ranked list. Users do not visit all documents in a ranked list. The common *examination assumption* [9] states that the deeper a user investigates a ranked list, the less likely they are to continue examining the list's next document; consistent with the probability ranking principle, deep down the ranked list we expect IR systems to return fewer relevant documents.

In our interaction model, at rank r, users decide either to continue examining another document, or to switch to a different sublist (cf. A3). The assumption that users are more likely to switch when they move deeper down the ranked list can be captured by controlling the parameter $p_{r,u}$ of the Bernoulli distribution with an exponential decay function:

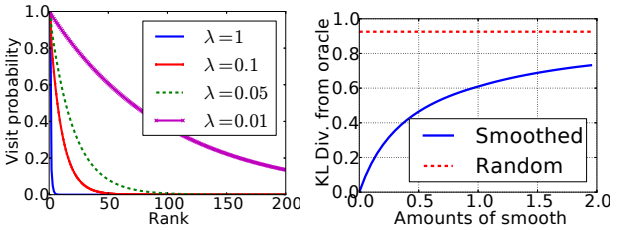
$$p_{r,u} = e^{-\lambda r}, \tag{3}$$

where λ controls the decay rate. That is, a user with a larger λ would decide to switch list at an earlier rank. The resulting exponential decay (Fig. 5(a)) is a good fit for Fig. 2(a).

We simulate the continuation decision of a user with λ_u in the following steps:

1. Compute $p_{u,r}$ with Eq. 3 given λ_u;
2. Draw a decision $s_{u,r} \sim Ber(p_{u,r})$.

Accurate and inaccurate users. When switching between sublists, some users make better choices than others. A good decision leads to a sublist with many (unseen) relevant documents ranked near the top, potentially reducing the total effort the user needs for



(a) Probability of reaching rank r given different λ.

(b) Amounts of smooth vs. KL-divergence from the oracle.

Figure 5: Illustration of parameter setup

his/her search task. We simulate users with different levels of accuracy by varying the *sublist selection probability* (cf. Section 3.1).

Specifically, we sample $\mathbf{c}_u$ from its conjugate prior distribution, i.e., a Dirichlet distribution $Dir(\alpha_u)$, with hyperparameters α_u. By setting α_u to different values we can simulate users with different types of prior knowledge. E.g., users who do not have a clue which sublist to choose can be simulated by setting α_u to a uniform distribution; an "accurate" user can be simulated by setting α_u proportional to the quality of the list, e.g., as measured by NDCG. The properties of the Dirichlet distribution ensure that the expected value of $c_{u,k}$ is $\alpha_k / \sum_j \alpha_j$, i.e., proportional to the performance of the sublists.

In summary, for a given α_u, we simulate a user's choice of sublist with the following procedure:

1. Draw $\mathbf{c}_u \sim Dir(K, \alpha_u)$;
2. Draw the decision vector $(f_1, \ldots, f_K) \sim Cat(K, \mathbf{c}_u)$.

Influence of user behavior on search performance. Before simulating system performance under varying conditions, we conduct a sanity check to examine how the above parameter settings influence estimated user effort given the following setup.

User task We consider an information gathering task where users aim to collect 1, 10, or *all* relevant documents;

Examination depth We set $\lambda \in \{1, .5, .1, .05, .01, .005, .001\}$ to reflect different user examination depths on a ranked list.

User accuracy We consider two cases: a uniform prior $\alpha_{u,k} = 1/K$, and a prior biased on list quality, i.e., α_k is set to the NDCG value of the corresponding result list.

User effort and gain As before, we assume equal effort for all actions and binary relevance to compute gain.

Given the possible settings above, we run the simulation over all combinations of these parameters, each for 1000 times.

Fig. 6 shows the simulation results. We plot the median of simulated user effort with two different user accuracy priors: NDCG vs. Random. With each prior, the continuation probability is set to different values with varying λ. From the figure we observe: (i) in all cases, irrespective of the value of λ and the task type, good prior knowledge about which sublist to choose is beneficial. A uniform prior corresponding to random selection always leads to more effort. (ii) The continuation probability has a limited effect when fewer relevant documents need to be found. When more relevant documents need to be found, e.g., in the Find-all task, it is better to go deeper down a ranked list (i.e., small λ). These observations are intuitive and provide a sanity check on the simulation results using our user interaction model and the proposed simulation strategy.

5.2 Analysis method

Next, we describe the method we employ to analyze conditions when an RLR interface is likely to be beneficial and when it is not.

5.2.1 Factors influencing RLR effectiveness

We identified the following factors that, presumably, determine whether an RLR interface is preferable over a basic one.

Query difficulty for the basic interface (D_q). A priori, if the ranked list in a basic interface is good, i.e., with all relevant documents on top, then users do not need to switch to other sublists for their tasks. We use the effort users need to accomplish a task with the basic interface ($effort_b$) as the indicator of D_q. The higher $effort_b$, the more difficult the query is for a basic interface.

Sublist relevance (R_q). The effectiveness of an RLR element should depend on the quality of the sublists created for result refinement. If the sublists would filter the relevant documents that were buried deep down in the original ranked list, then it is likely to help users to accomplish their tasks faster. We compute R_q as the averaged NDCG scores over the sublists of a query.

Sublist entropy (H_q). A priori, if few sublists cover most of the relevant documents, these could help to effectively filter out irrelevant documents. Meanwhile, if many sublists contain many relevant documents, then it may be easy for users to find them. In short, we believe the effectiveness of an RLR system is related to how the relevant documents are distributed among sublists, but the exact relation is yet to be explored.

We compute $H_q = -\sum_i p_i \log(p_i)$ as the sublist entropy of a query q, where p_i is the probability that the sublist i of query q contains relevant documents, derived from the empirical distribution of the relevant documents among the sublists of q.

User accuracy (U_{level}). As shown in Fig. 6, how users choose the sublists makes a big difference on the effectiveness of using RLR. Following the distribution of NDCG scores of the sublists is a good strategy, while choosing randomly leads to inferior performance.

Here, we aim to investigate the impact of user accuracy at a more refined level, i.e., how accurate should the user be in order to make the RLR work? Recall that user sublist selection behavior is controlled by the parameter α. Assuming user choices following NDCG scores are "oracles," by gradually smoothing out α with respect to this oracle distribution, we can create user accuracies of different levels between the oracle and the complete random choices (i.e., uniform α). Fig. 5(b) shows the relation between amounts of smoothing added and the median of the KL divergence of the "new" user from the "oracle" user over the 50 test topics. We create 4 user levels, with the amount of smoothing set to 0, 0.1, 0.5, and 1.0, corresponding to the oracle user (level 1), and approximately 15% (level 2), 50% (level 3), and 67% (level 4) less accurate users.

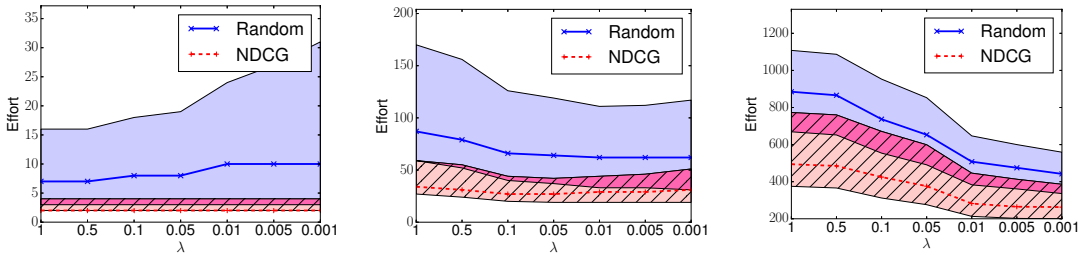
User task. Intuitively, the impact of these factors would be different with respect to different user tasks. For example, when the task is to find all relevant documents, it is not important how good the top of the ranked list is, but rather, where the last relevant document is located. We consider finding 1, 10, and *all* relevant documents.

As a final note, in Fig. 6 we have observed the influence of user search depth on the RLR performance. Overall its impact is not as obvious as user accuracy, in terms of the magnitude of changes in efforts it leads to. In this analysis, we focus on the sublist selection aspect of the users and fix λ to 0.01, which seems to be optimal for finding all relevant documents. For the other two tasks it does not seem to make a major difference when set to a different value.

5.2.2 Analyzing the impact of the identified factors

Let $\Delta_{effort} = effort_b - effort_r$ be the difference between the effort needed to complete a search task with a basic interface and that with an RLR interface. We then make $\Delta_{effort} > 0$ a dependent variable (DV), which takes binary values (1 as yes, 0 as no), and the above four factors (D_q, R_q, H_q, and U_{level}) as independent variables (IVs). Our goal is to investigate how each of the IVs, and their interactions, influence the outcome of whether or not an RLR interface improves over a basic interface.

We apply a generalized linear model (GLM) for this purpose, which allows us to analyze how the IVs contribute to explain the variance observed in the DV. Specifically, given that $\Delta_{effort} > 0$ is a binary variable, we take the form of a logistic regression model. We fit the models with the data simulated with parameters set to different user tasks and different U_{levels}. The conditions w.r.t. the rest of the variables D_q, R_q, and H_q are determined at a per-query level, which comes with the test collection.



(a) Task: find - 1 (b) Task: find - 10 (c) Task: find - all

Figure 6: Influence of user patience and their accuracy in choosing result lists. The line curves represent the median of the simulation results, and the shades indicate their upper and lower quartiles.

To determine which IVs and interaction terms should be included in the model, we conducted model selection based on the Bayesian information criterion (BIC) using both forward and backward selection. Further, we expect that for different types of tasks (e.g., Find-1 vs. Find-all), the importance of these factors would differ dramatically, and different models would be appropriate. Therefore, we fit a model for each of the user tasks individually.

5.3 When does RLR help?

Table 1 shows the parameters for models best able to predict whether the RLR interface will be effective based on combinations of the four factors for each task.

5.3.1 Main effect

We observe that for Find-1 none of the main factors in the model have a significant effect on the dependent variable (RLR effectiveness). However, for Find-10 and Find-all tasks we see that the U_{level} has a significant effect. The negative coefficients for the U_{level} variables indicate that, as users deviate from the "oracle" sublist selection behavior, the log-odds of the RLR interface being effective decrease. Those user level effects for Find-10 and Find-all but not Find-1 suggest that if a user's task is to locate 1 document, then just the accuracy with which users select sublists is not enough to predict whether an RLR interface will be beneficial. One explanation is that, since most sublists will have at least one relevant document ranked highly, users do not need to be accurate in their choice of sublist to achieve the task. When collecting more relevant documents however, knowing which sublist to pick *is* important.

Regarding R_q we find that it has a significant effect only for Find-all. As the average relevance of sublists increases the log-odds of the RLR interface being effective increase as well. That is, having sublists with relevant documents ranked high is essential for the RLR interface to be effective for the Find-all task. For the Find-1 and Find-10 tasks sublist relevance alone is not enough to predict RLR effectiveness and the effect depends on the interaction between two or more of the main factors. We look into these interaction terms in more detail next.

Table 1: Estimated coefficients of the selected models and their effects on the odds that an RLR interface helps. The overall effect of user accuracy (ulevel) is tested by Wald test. The model goodness-of-fit (GOF) is tested by Hosmer-Lemeshow test. Significance codes: $\leq$0.001 ($^{\blacktriangle}$); $\leq$0.01($^{\triangle}$); $\leq$ 0.05(*).

Coefficients	Find-1	Find-10	Find-all
(Intercept)	-7.3401	-10.4365	-0.5337
D_q	0.1058	-0.0686	0.0017
U_{level2}	3.2234	-2.1309$^{\blacktriangle}$	-5.1061$^{\blacktriangle}$
U_{level3}	1.5590	-5.5278$^{\blacktriangle}$	-8.0140$^{\blacktriangle}$
U_{level4}	-2.3189	-8.1936$^{\blacktriangle}$	-8.0140$^{\blacktriangle}$
H_q	-1.0443	3.6347	-1.6492
R_q	–	-49.7916	114.9398$^{\blacktriangle}$
$D_q : U_{level2}$	-1.6547	–	–
$D_q : U_{level3}$	-2.0041*	–	–
$D_q : U_{level4}$	-2.0683	–	–
$D_q : H_q$	1.3103$^{\blacktriangle}$	-0.0968	–
$D_q : R_q$	–	3.2363$^{\triangle}$	0.0906*
$H_q : R_q$	–	13.9680	-57.2773$^{\blacktriangle}$
$D_q : H_q : R_q$	–	-0.8422*	–
Overall effect of U_{level}	χ^2=1.6 (*df=3*)	$\chi^2 = 16.0^{\triangle}$ (*df=3*)	$\chi^2 = 25.6^{\blacktriangle}$ (*df=3*)
Model GOF (p-value)	0.9339	0.9928	0.9213

Regarding R_q we find that it has a significant effect only for Find-all. As the average relevance of sublists increases the log-odds of the RLR interface being effective increase as well. That is, having sublists with relevant documents ranked high is essential for the RLR interface to be effective for the Find-all task. For the Find-1 and Find-10 tasks sublist relevance alone is not enough to predict RLR effectiveness and the effect depends on the interaction between two or more of the main factors. We look into these interaction terms in more detail next.

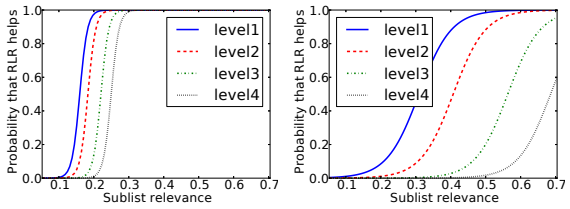
5.3.2 Effect of interaction terms

To investigate the effect of the interaction terms on the probability of RLR effectiveness, we express the relation visually. Due to space limitation, we focus here on the model for the Find-10 task.

To visualize interaction terms between continuous variables, we plot the predicted value of the DV against one varying IV, and re-center the other IVs to a fixed level. We take the 25% and 75% quantile of the values of a variable as its low and high level, respectively. A model fitting the re-centered data then shows the effect of the varying IV on the DV with respect to the different levels of the re-centered IVs. For Find-10 there are two interactions terms that have significant effect: D_q:R_q and D_q:R_q:H_q. For D_q:R_q, we re-center D_q to its low and high levels, and fix H_q to its median. For D_q:R_q:H_q we re-center both D_q and R_q to a low and high level.

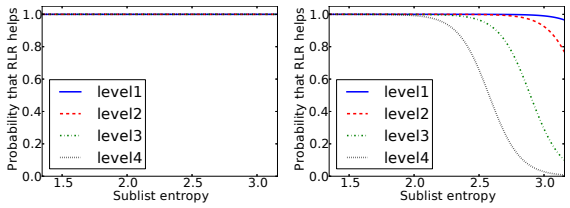
Fig. 7(a) shows the effect of increasing R_q on the probability $P(\Delta_{effort} > 0)$ when D_q is high for varying levels of user accuracy. At relatively low levels (0.15 to 0.25) of R_q there is a steep increase in the probability of RLR being more effective for all user levels. This suggests that when a query is difficult (i.e., the quality of the original ranked list is low), the sublists and the users do not need to be very accurate for an RLR interface to be more effective than a basic interface. In Fig. 7(b) we see that when query difficulty is low, high quality sublists (relevant documents ranked high) and higher user accuracy are necessary for an RLR interface to be helpful.

The relation between D_q, R_q, and H_q in the final interaction term for task Find-10 is shown in Fig. 8. It shows the effect of increasing H_q on $P(\Delta_{effort} > 0)$ under conditions in terms of combinations of different levels of D_q (high/low), R_q (high/low), and varying levels of user accuracy. We see that a high level of D_q combined with low/medium levels of H_q result in a relative high $P(\Delta_{effort} > 0)$ for both high (Fig. 8(c)) and low (Fig. 8(a)) levels of R_q. This interaction aligns with our intuition of when sublists

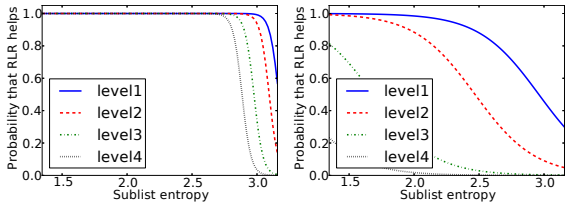


(a) D_q: high; H_q: median (b) D_q: low; H_q: median

Figure 7: Effect of interaction terms query difficulty : sublist relevance for task Find-10.



(a) D_q: high; R_q: high (b) D_q: low; R_q: high



(c) D_q: high; R_q: low (d) D_q: low; R_q: low

Figure 8: Effect of interaction terms query difficulty : sublist relevance : sublists entropy for task Find-10.

are beneficial, i.e., having a single high quality sublist when the original ranked list is of low quality. When D_q is low, we observe that $P(\Delta_{effort} > 0)$ decreases at low levels of H_q when R_q is low (Fig. 8(d)) and at medium levels of H_q when R_q is high (Fig. 8(b)). This suggests that at lower levels of query difficulty very specific conditions need to be met for an RLR interface to be beneficial.

H_q plays a role in different interaction terms depending on the task as well (cf. Table 1). For Find-1 we find that as both D_q and H_q increase the log-odds of the RLR interface being beneficial increases. Since the task requires a single relevant document, the ranking within the vertical is less important. Having more sublists with a relevant document allows users to complete the task effectively with the RLR interface even when users select sublists randomly. For Find-all, an increase in both H_q and R_q result in a decrease in the log-odds of the RLR interface being beneficial. As we saw in Fig. 8, for Find-10 high sublist entropy results in low probability of an RLR interface being helpful. Sublist relevance only plays a role when query difficulty is low; however, as the number of relevant documents to be found increases, it is less likely that enough documents are available at the top of the ranking.

5.3.3 *Summary*

Query difficulty alone is not a good predictor for the probability of an RLR interface being helpful. Depending on the task, different factors determine whether users will be more effective with the basic or RLR interface. In the case of Find-1, sublists do not have to be of high quality for an RLR interface to be helpful; it becomes more likely to be beneficial when the query is difficult and the entropy of the sublists is high. For Find-10, high query difficulty and low entropy are conditions for the RLR interface to be beneficial. The importance of sublist relevance depends on user accuracy; when they are accurate, lower levels of sublist relevance are necessary. For Find-all, the conditions necessary for an effective RLR interface are high query difficulty, high user accuracy, high sublist relevance and low entropy.

6. RELATION TO TRADITIONAL METRICS

We have illustrated how our evaluation framework can be used for simulating and predicting RLR system performance in two ways, and its efficacy has been validated with usage data. Next, we discuss how it relates to metrics for evaluation of traditional search systems, i.e., normalized Discounted Cumulative Gain [19, nDCG], Expected Reciprocal Rank [6, ERR], normalized Rank-biased precision [28], average precision (AP), and precision@10 (P@10).

From a modeling perspective, Carterette [4] has proposed a conceptual framework for analyzing and comparing different effectiveness measures (for traditional systems). Our framework is close to the category of Model 3 under his classification, i.e., computing the effort a user needs to achieve a particular amount of utility. Further, all metrics discussed in [4] compute an expected value (utility, effort or gain). Computing the expected performance directly is rather intractable in our setting, as the order in which sublists are selected and the number of results viewed in each of these lists is non-deterministic. To compute the expected performance, one needs to obtain the distribution of all possible orders in which sublists are selected. Here, simulation provides samples of possible sequences from which performance can be approximated (Section 3.3).

We now move on to an empirical investigation. By examining to what extent metrics for traditional systems are able to predict the performance of an RLR system, we investigate whether our framework offers new insights. Table 2 shows the correlation between traditional measures and actual user effort (column 1, 2; obtained in Section 4), predicted user effort (column 3, 4), and the difference between user effort with the basic and RLR interface ($\Delta effort_u$ in column 5, 6). We observe that nDCG measured at low cut-offs (10) has no significant correlation with user or predicted effort. When all relevant documents are taken into account (nDCG@all), the correlation is significant at $\rho = -.42$ (negative, for gain vs. effort). We observe a similar pattern for binary nDCG (BnDCG); however, in this case the correlation is stronger than for nDCG. When collecting usage data we did not distinguish between highly relevant and relevant documents. Other measures have a negative correlation with user effort and our model in the range between nDCG and BnDCG. We focus on BnDCG here, as it is most strongly correlated.

The correlation between user effort and BnDCG@all is $\rho = -.72$, which indicates that for topics with high BnDCG@all scores, i.e., with many relevant documents at the top of the ranking, user effort is low. The magnitude of the correlation of both BnDCG and our model with actual user effort is high. The negative correlation between BnDCG@all and our model is lower than that of either measure with user effort ($\rho = -0.59$, $p < 0.001$), indicating that these measures disagree on the effort needed for some topics.

We apply BnDCG@all to the task of predicting whether a topic would cost a user more effort with the basic or the RLR interface ($\Delta effort_u$). We compute the correlation between BnDCG@all and $\Delta effort_u$. Results are listed in Table 2 (last two columns). There is no significant correlation ($\rho = 0.04$, $p = 0.776$): BnDCG@all cannot differentiate between topics suitable for a basic or RLR interface. In comparison, the correlation of $\Delta effort_u$ with the pre-

Table 2: Correlation of traditional metrics with user effort, predicted effort, and the $\Delta effort_u$ (cf. Fig. 4).

	user effort		simulated effort		$\Delta effort_u$	
measure	ρ	p-value	ρ	p-value	ρ	p-value
nDCG@10	-0.21	0.142	-0.19	0.185	0.02	0.896
nDCG@all	-0.42	0.002	-0.34	0.016	0.00	0.994
NRBP	-0.41	0.003	-0.33	0.018	0.08	0.568
ERR@10	-0.45	0.001	-0.36	0.010	0.08	0.567
P@10	-0.56	<0.001	-0.46	<0.001	0.00	0.980
AP	-0.63	<0.001	-0.54	<0.001	0.02	0.875
BnDCG@10	-0.54	<0.001	-0.44	0.001	0.02	0.875
BnDCG@all	**-0.72**	**<0.001**	**-0.59**	**<0.001**	0.04	0.776
our model	**0.79**	**<0.001**	–	–	**-0.49**	**<0.001**

dicted effort (by our model) is significant ($\rho = -0.49, p < 0.001$). That is, simulated effort tells us which interface is the best.

As a final remark, the lack of correlation between traditional metrics and $\Delta effort_u$ confirms our observation that query difficulty, i.e., the quality of the original ranked list alone, is not sufficient to predict whether an RLR interface is preferable over a basic interface (cf. Section 5.3).

7. CONCLUSION

We have developed a simulation-based evaluation framework that measures the effectiveness of systems enabling result refinement, e.g., facets or filters. Its key component is an interaction model that characterizes the user's search behavior in the presence of result list refinement features. Using this framework, we investigate whole system performance, under various conditions. Instantiating the parameters of the user interaction model, corresponding to properties of search task and user type, allows us to predict system performance for specific groups of users. We validated the predictions made using data collected with two search systems, re-using the TREC Federated Search test collection.

We found that user effort estimated by our model is correlated significantly with actual user effort measured in the user data. We applied our interaction model to the task of predicting when a user should or should not use an RLR interface, and found a significant correlation between the predictions we made and observations in the user data. We did not find such correlations when applying traditional retrieval metrics to this task, demonstrating the value of the proposed user interaction model for search with result refinement.

Our study extends user interaction models beyond the classic "10 blue links." It provides a means to evaluate retrieval systems while considering the interaction effects between non-standard search UI features and search system effectiveness and the variability in how different people use the search UI.

Acknowledgements. This research was partially supported by the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement nr 312827 (VOX-Pol), the Netherlands Organisation for Scientific Research (NWO) under project nrs. 727.011.005, 612.001.116, HOR-11-10, 640.006.013, 612.066.930, CI-14-25, SH-322-15, Amsterdam Data Science, the Dutch national program COMMIT, the ESF Research Network Program ELIAS, the Elite Network Shifts project funded by the Royal Dutch Academy of Sciences (KNAW), the Netherlands eScience Center under project nr. 027.012.105, the Yahoo! Faculty Research and Engagement Program, the Microsoft Research PhD program, the HPC Fund, and the Dutch Technology Foundation (STW) under project nr. 13675.

8. REFERENCES

[1] M. Agosti, N. Fuhr, E. Toms, and P. Vakkari. Evaluation methodologies in information retrieval dagstuhl seminar 13441. In *ACM SIGIR Forum*, volume 48, pages 36–41. ACM, 2014.

[2] L. Azzopardi. Modelling interaction with economic models of search. In *SIGIR'14*, pages 3–12, 2014.

[3] M. Bron, J. Van Gorp, F. Nack, L. B. Baltussen, and M. de Rijke. Aggregated search interface preferences in multi-session search tasks. In *SIGIR'13*, pages 123–132. ACM, 2013.

[4] B. Carterette. System effectiveness, user models, and user utility: a conceptual framework for investigation. In *SIGIR'11*, 2011.

[5] O. Chapelle and Y. Zhang. A dynamic bayesian network click model for web search ranking. In *WWW'09*, pages 1–10, 2009.

[6] O. Chapelle, D. Metlzer, Y. Zhang, and P. Grinspan. Expected reciprocal rank for graded relevance. In *CIKM'09*, 2009.

[7] A. Chuklin, P. Serdyukov, and M. de Rijke. Click model-based information retrieval metrics. In *SIGIR '13*, 2013.

[8] A. Chuklin, I. Markov, and M. de Rijke. *Click Models for Web Search*. Morgan & Claypool Publishers, 2015.

[9] N. Craswell, O. Zoeter, M. Taylor, and B. Ramsey. An experimental comparison of click position-bias models. In *WSDM'08*, 2008.

[10] T. Demeester, D. Trieschnigg, D. Nguyen, and D. Hiemstra. Overview of the trec 2013 federated web search track. In *TREC'14*.

[11] D. Downey, S. Dumais, and E. Horvitz. Models of searching and browsing: languages, studies, and applications. In *IJCAI'07*, 2007.

[12] G. Dupret and B. Piwowarski. A user browsing model to predict search engine click data from past observations. In *SIGIR '08*, 2008.

[13] J. English, M. Hearst, R. Sinha, K. Swearingen, and K.-P. Yee. Flexible search and navigation using faceted metadata. In *University of Berkeley*. Citeseer, 2002.

[14] N. Fuhr. A probability ranking principle for interactive information retrieval. *Information Retrieval*, 11(3):251–265, 2008.

[15] F. Guo, C. Liu, and Y. Wang. Efficient multiple-click models in web search. In *WSDM '09*, 2009.

[16] J. He, M. Bron, and A. P. de Vries. Characterizing stages of a multi-session complex search task through direct and indirect query modifications. In *SIGIR'13*, pages 897–900, 2013.

[17] J. He, M. Bron, L. Azzopardi, and A. de Vries. Studying user browsing behavior through gamified search tasks. In *GamifIR'14*, pages 49–52, 2014.

[18] J. Huang, R. W. White, and S. Dumais. No clicks, no problem: using cursor movements to understand and improve search. In *SIGCHI'11*.

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

[20] K. Järvelin, S. L. Price, L. M. Delcambre, and M. L. Nielsen. Discounted cumulated gain based evaluation of multiple-query IR sessions. In *ECIR'08*, pages 4–15, 2008.

[21] T. Joachims, L. Granka, B. Pan, H. Hembrooke, and G. Gay. Accurately interpreting clickthrough data as implicit feedback. In *SIGIR'05*, pages 154–161. ACM, 2005.

[22] E. Kanoulas, B. Carterette, P. D. Clough, and M. Sanderson. Evaluating multi-query sessions. In *SIGIR'11*, 2011.

[23] A. Kashyap, V. Hristidis, and M. Petropoulos. Facetor: cost-driven exploration of faceted query results. In *CIKM '10*, 2010.

[24] W. Kong and J. Allan. Extending faceted search to the general web. In *CIKM'14*, 2014.

[25] J. Koren, Y. Zhang, and X. Liu. Personalized interactive faceted search. In *WWW '08*, pages 477–486, 2008.

[26] R. L. Kumar, M. A. Smith, and S. Bannerjee. User interface features influencing overall ease of use and personalization. *Information & Management*, 41(3):289–302, 2004.

[27] C. Li, N. Yan, S. B. Roy, L. Lisham, and G. Das. Facetedpedia: dynamic generation of query-dependent faceted interfaces for wikipedia. In *WWW '10*, pages 651–660. ACM, 2010.

[28] A. Moffat and J. Zobel. Rank-biased precision for measurement of retrieval effectiveness. *ACM Trans. Inf. Syst.*, 27(1):Article 2, 2008.

[29] P. Morville and J. Callender. *Search patterns*. O'Reilly Media, 2010.

[30] A. Schuth and M. Marx. Evaluation methods for rankings of facetvalues for faceted search. In *CLEF'11*, pages 131–136, 2011.

[31] M. D. Smucker and C. L. Clarke. Stochastic simulation of time-biased gain. In *CIKM'12*, pages 2040–2044, 2012.

[32] D. Vandic, F. Frasincar, and U. Kaymak. Facet selection algorithms for web product search. In *CIKM '13*, pages 2327–2332. ACM, 2013.

[33] E. Voorhees, D. K. Harman, et al. *TREC: Experiment and evaluation in information retrieval*. MIT, 2005.

[34] B. S. Wynar, A. G. Taylor, and J. Osborn. *Introduction to cataloging and classification*. Libraries Unlimited, 1985.