Simulating User Selections of Query Suggestions

Jiepu Jiang
School of Information Sciences,
University of Pittsburgh
jiepu.jiang@gmail.com

Daqing He
School of Information Sciences,
University of Pittsburgh
dah44@pitt.edu

ABSTRACT

In this paper, we identify that simulating user selections of query suggestions is one of the essential challenges for automatic evaluation of query suggestion algorithms. Some examples are presented to illustrate where the problem lies. A preliminary solution is proposed and discussed.

1. MOTIVATION

Recent studies [3, 11] found that user ratings of query quality are very different from those evaluated using system-oriented metrics (e.g. nDCG@10 of query results). According to these findings, it is very likely that users may not be able to identify the good queries when provided with a list of query suggestions. Therefore, we believe that user selections of query suggestions should be considered into the evaluation of query suggestion algorithms, so as to better understand the effectiveness of query suggestions for practical systems and users.

Existing methods either did not consider this issue or adopted unwarranted assumptions. For example, and Dang et al. [1] evaluated a list of query suggestions by the performance of the best query, which implicitly assumed that users can make perfect judgments and adopt the best query. Sheldon et al. [10] evaluated by the average performance of the suggested queries, assuming that users randomly select query suggestions. To illustrate where the problem lies, we present two examples where the existing methods show their limitations.

Example 1. *How many suggested queries should be presented?*

With more queries being presented, the maximum performance of the queries will probably be enhanced. However, the average performance of the queries will probably decline, because query suggestion algorithms are designed to rank good queries at higher positions. In such case, the existing evaluation methods come to conflict conclusions. In order to determine the number of queries to be presented, we need to investigate the target user groups on their ability of judging queries. It is more likely for users with high judge ability to avoid selecting ineffectively queries and benefit from a long list of query suggestions, while for users with low judge ability, increasing the number of query suggestions may lead to decline of search performance.

Example 2. *Should we present query suggestions or not?*

Query suggestions can lead to decline of search performance if users adopt query suggestions that underperform those could be reformulated by users themselves. If the quality of the user's own query reformulations lies between the average and the best performance of the suggested queries, the query suggestions are very likely beneficial for users with high judge ability but risky for those with low judge ability. In addition, the performance of query suggestions also depends on the quality of the users' own reformulations. As Kelly et al. found [7], "query suggestions seem to have an advantage when subjects face a cold-start problem and when they exhaust their own ideas for searches", where the users cannot reformulate effective queries.

The rest of the paper discusses a preliminary solution.

SIGIR 2013 Workshop on Modeling User Behavior for Information Retrieval Evaluation (MUBE 2013), August 1, 2013, Dublin, Ireland.

2. METHODS

We are particularly interested in a specific time of a search session when a user has issued several queries and is now offered a ranked list of query suggestions. The user can either adopt one suggested query or issue his/her own reformulation for the next round of search. We believe that an evaluation model for query suggestion should examine how helpful a suggested query is for the user at that specific time of the session. We use the following notations in further discussions:

$S\{q_1, \dots, q_n\}$	A list of n query suggestions presented to the user.
C	A candidate set of query suggestions actually judged by the user. C may include only parts of the queries in S.
q_0	The user's own query reformulation in mind, i.e. the user will issue q_0 if no query suggestions are offered.
q'	The actual query adopted by the user.
$u(q)$	A measure for the utility of the query q.

The evaluation can be achieved by calculating the expected difference between the utility of the follow-up searches with and without the query suggestions, as shown in the left side of Eq(1). Here we simply assume that $u(q_0)$ is a constant provided by evaluation datasets, so that the main challenge is to measure $E(u(q'))$, as shown in the right side of Eq(1).

$$E\left(u\left(q'\right)-u\left(q_0\right)\right)=E\left(u\left(q'\right)\right)-u\left(q_0\right) \quad (1)$$

Apparently, the optimized performance will be achieved if q' is the best query (by the utility measure u) among S and q_0. However, the query actually adopted by the user may not be the real best one. First, the user may not be persistent enough to judge all the query suggestions, since it takes time and effort to judge each query. Second, the user may not be able to identify the best query among those being judged. This is because that the user's perceived utility of queries may be different from the actual utility since the user does not know the search results when judging queries.

Therefore, we further calculate $E(u(q'))$ using a two-step approach as shown in Eq(2). First, we calculate the probability that the user will judge a possible subset S' of S, i.e. $P(S'|S)$. Second, we calculate the probability that the user will select a query q_i among q_0 and S' for the next round of search, i.e. $P(q'=q_i|q_0, S')$. Finally, we marginalize over all possible selected queries to come out the expected utility of the selected query.

$$E\left(u\left(q'\right)\right)=\sum_{S'\subset S}P\left(S'\mid S\right)\cdot\sum_{q_i\in\{q_0,S'\}}P\left(q'=q_i\mid q_0,S'\right)\cdot u\left(q_i\right) \quad (2)$$

2.1 Users' Judging Persistence

To estimate $P(S'|S)$, we adopt a model similar to the browsing model in rank-biased precision (RBP) [9]. We assume that the user judges queries in S by the sequences of the queries ranked by the system. The user will always judge the first query in S. After judging a query, the user has the probability p_{next} to continue judging the next one, or $1 - p_{\text{next}}$ to stop and select the believed best query among those having been judged so far. Suppose S_k is $\{q_1, q_2, \dots, q_k\}$ ($k \le n$), i.e. the user has judged this subset of queries and stops after judging the kth query q_k, we can calculate $P(S_k|S)$ as Eq(3):

$$P\left(S_k\{q_1,q_2,...,q_k\}\mid S\right)=(1-p_{next})\cdot p_{next}^{k-1} \quad (3)$$

2.2 Users' Judging Ability

For a judged set of queries S' (in which all queries are judged), we assume that the user will compare each query in S' as well as the user's own query q_0 to select a believed best one. We use C for the set of candidate queries, including q_0 and S'. We have the following intuitions regarding query selection:

> ***Intuition 1*** The better the user's judging ability is, the more likely that the user can select the best query in C.
>
> ***Intuition 2*** The better a query's quality is, the more likely that the user will select the query.

We adopt a parameter p_{judge} for the user's judging ability. p_{judge} is defined as the probability of making a correct pairwise judgment on the utility of two queries (i.e. making a correct statement on which query has the higher utility). $p_{judge} = 0.5$ indicates that the user's judgments are no better than random selection; $p_{judge} > 0.5$ indicates that the user has a general capability of judging queries (the higher the value of p_{judge}, the better the user's judging ability); $p_{judge} < 0.5$ indicates that the user's judgments are opposite to those measured by u.

We assume that users select their believed best queries through a round-robin tournament process involving pairwise judgments over all possible pairs of candidate queries, as shown in Table 1. We executed the simulation tournament for a large number of times and recorded down the outcome of each iteration. Finally, we estimate the probability of selecting a query in C by the maximum likelihood estimation that the query was selected.

Table 1. Query selection tournament algorithm.

Algorithm: query-selection-tournament
Input: $C\{q_1, q_2, \dots, q_m\}$
Output: one selected query
init array *scores*[1…m] // stores the scores of the m queries
init array *winner* // stores the selected query/queries
for i from 1 to $m - 1$
  for j from $i + 1$ to m
    q_x = believed better one of q_i and q_j (a random factor with the probability p_{judge} being the actual better one of q_i and q_j)
    scores[x]++ // the judged better query will earn 1 point
winner = the query(s) with the highest score in *scores*
if the *winner* array contains only one query
  return the query in *winner*
else // has more than one "winners"
  return tournament-query-selection(*winner*) // recursion

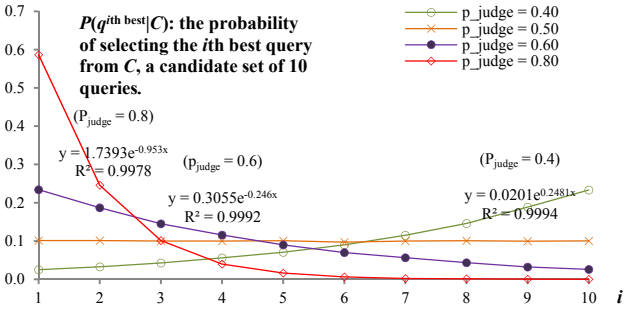

Figure 1. Estimated probability of $P(q^{(ith\ best)}|C)$.

When a subset S' is determined, the probability of selecting a query in S' does not depend on the original rank of the query in S', but the rank of the query in $C\{q_0, S'\}$ by the utility measure $u(q)$. p_{judge} and the number of queries in C can also affect the selection probability. Let C be a set of 10 queries. Figure 1 shows the probability of selecting the ith best query in C, i.e. $P(q^{ith\ best}|C)$, estimated after running the tournament 100,000 times. Although we have not derived a formula for the query selection probability, we noticed that it can be fitted by an exponential function. Figure 1 shows several examples. The trend lines fit the observed results perfectly, with $R^2 > 0.99$.

2.3 Experiment Settings

Dataset. Our model uses the user's own query reformulation for evaluation, which can be provided from a session search dataset [6]. For a static session $q^{(1)}, q^{(2)}, \dots, q^{(m)}$, we can evaluate query suggestions for $q^{(x)}$ ($x < m$) so that the dataset can provide information on $q^{(x+1)}$.

Utility measures. The utility measures adopted in previous works include those based on human ratings [8], search logs [2], and the session-level evaluation metrics [4–6]. However, further studies are required to verify their validity in the evaluation of query suggestion algorithms.

3. DISCUSSION AND FUTURE WORKS

According to Figure 1, users with high judge ability are very likely to be able to identify comparatively better queries. For example, when $p_{judge} = 0.8$, the user has over 80% probability to select the top 2 best queries. However, when $p_{judge} = 0.6$ (a comparatively low judge ability), the user has about 30% chances to select below average quality queries. This suggests two strategies of query suggestions for different user groups: for users with high judge ability, the algorithm can aim at increasing the upper bound quality of the suggested queries (finding the best possible query suggestion); for users with low judge ability, it is rather risky to return ineffective queries and therefore the algorithms should balance between "finding the best possible query suggestion" and "maintaining the overall quality of the query suggestions".

Our future work will focus on the verification and refinement of the proposed model through user experiments, including mainly the validation of utility measures and the following issues:

- The current model did not consider the modeling of p_{judge}. Therefore, one of our future works is to properly measure p_{judge} from user experiments and model p_{judge} based on various factors. Except for user factors, p_{judge} may also be affected by the two queries being compared.
- As reported in [7], users "felt that the query suggestion system helped them in a variety of ways, some of which are not detectable from the log". The current model assumes that the users' own query reformulations will not be affected by the query suggestions. However, it is possible that the suggested queries may also influence users' own query reformulations.

4. REFERENCES

[1] Dang, V. et al. 2010. Query reformulation using anchor text. In WSDM'10.

[2] Duarte Torres, S. et al. 2012. Query recommendation for children. In CIKM'12.

[3] Hauff, C. et al. 2010. A comparison of user and system query performance predictions. In CIKM'10.

[4] Järvelin, K. et al. 2008. Discounted Cumulated Gain Based Evaluation of Multiple-Query IR Sessions. In ECIR'08.

[5] Jiang, J. et al. 2012. Contextual evaluation of query reformulations in a search session by user simulation. In CIKM'12.

[6] Kanoulas, E. et al. 2011. Evaluating multi-query sessions. In SIGIR'11.

[7] Kelly, D. et al. 2009. A comparison of query and term suggestion features for interactive searching. In SIGIR'09.

[8] Ma, Z. et al. 2012. New assessment criteria for query suggestion. In SIGIR'12.

[9] Moffat, A. et al. 2008. Rank-biased precision for measurement of retrieval effectiveness. ACM Transactions on Information Systems, 27, 1.

[10] Sheldon, D. et al. 2011. LambdaMerge: Merging the Results of Query Reformulations. In WSDM'11.

[11] Wu, W. et al. 2012. User evaluation of query quality. In SIGIR'12.