Pitt at TREC 2013: Different Effects of Click-through and Past Queries on Whole-session Search Performance

Jiepu Jiang*
School of Computer Science,
University of Massachusetts Amherst
jpjiang@cs.umass.edu

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

ABSTRACT

Past search queries and click-through information within a search session have been heavily exploited to improve search performance. However, it remains unclear how do these two data source contribute to whole-session search performance due to the lack of reliable evaluation approaches. For example, as pointed out in our last year's report [2], using past search queries as positive relevance feedback information can make search results of the current query similar to previous queries' results. Such issues cannot be disclosed by evaluation metrics such as nDCG@10.

Therefore, in this paper, we focus on analyzing the effects of past queries and click-through information on whole-session search performance. We adopted alternative evaluation approaches other than the TREC official ones. We found that past queries may seemingly enhance nDCG@10 by retrieving previously returned results, which is difficult to result in real improvements of whole-session search performance; in comparison, click-through can enhance search performance without sacrificing search novelty, consequently leading to improved search performance across the whole session. However, after appropriate demotion of repeated results, both past queries and click-through can improve search performance while balancing novelty of results.

Keywords

Search session; TREC; evaluation; relevance feedback.

1. INTRODUCTION

Since 2011, the main task of the TREC session track is to improve search performance of a query in a search session using previous user behaviors (such as past search queries and clicked results). However, our studies [1, 2] revoked us to reconsider the validity of this task and its evaluation approach. In both the 2011 and 2012 TREC session track logs, we found that each time reformulating a query, there was no significant change of search performance (as measured by nDCG@10), but substantial difference in the set of results retrieved (with the Jaccard similarity of two queries' top 10 results ranging from 0.1 to 0.2). Due to this fact, we argued that users may expect to find novel search results, instead of simply to improve search performance when they reformulate queries [2]. Therefore, it may also be problematic to evaluate a system purely by whether or not it can improve search performance of a query in a search session and the magnitude of the improvement.

Except for the reason stated above, our concern also comes from the observation that a popular approach taken by many groups in the TREC session tracks may sacrifice novelty to improve search performance. The approach is to utilize past queries and clicked results as positive relevance feedback information for the current query. Our systems in the 2011 and 2012 TREC session tracks [2, 3] as well as many other participants' systems all adopted similar approaches. However, previous search queries and clicked results are probably related to relevant but obsolete information for the users. As found in our last year's TREC report [2], search results became more similar to those retrieved by previous queries after applying this approach. Therefore, it is unclear whether or not the improvement of search performance comes from novel relevant information or just some repeated relevant results found also by previous queries. Such issues had not been disclosed in the TREC session track due to its evaluation approaches.

These two reasons motivate us to explore approaches balancing search performance and novelty of results in a search session, as well as alternative metrics for evaluating systems and analyzing search results at whole-session level. We focus on the following research goals in this year's participation:

1. Although last year we suspected that the positive relevance feedback approach is problematic, we did not fully investigate whether it is true and how serious it is due to the lack of proper evaluation approaches. Also, it is unclear how such issues affect system performance at whole-session level. Therefore, this year we conducted more detailed analysis of this approach based on alternative metrics indicating whole-session search performance. We introduce this approach and our findings in section 2 and 3. Results indicate that using previous queries as positive relevance feedback information may largely reduce the novelty of search results to enhance metrics such as nDCG@10. In comparison, using click-through as positive relevance feedback consistently improves search performance without loss of novelty of results.

2. Last year we proposed an approach to demote the rankings of the repeatedly occurred results. However, this approach only considered the results repetition issue but cannot help with the content novelty in a search session. Therefore, we propose another approach trying to rank results based on whole-session relevance (run "KM1" and "KM1N"). Unfortunately, it did not perform well in the TREC 2013 session track evaluation. Due to the lack of qrels data at the time of finalizing our report, we do not analyze this approach in details.

The rest of the paper introduces our approaches, experiments, and findings.

2. POSITIVE RELEVANCE FEEDBACK USING PAST SEARCH BEHAVIORS

This section addresses our first research goal this year. We evaluate how a popular approach of using past search behaviors as positive relevance feedback affect whole-session search performance.

* The majority of the work was finished when Jiepu Jiang was at School of Information Sciences, University of Pittsburgh.

2.1 Retrieval Model

The approaches adopted by the Pitt group in the 2011 and 2012 TREC session track [2, 3] are variants of the "context-sensitive relevance feedback" approach [8]. This approach adopts the KL-Divergence language modeling framework [5, 9] for retrieval and highlights a query language model combining the current search query, past search queries, and click-through documents. In our study, we adopt this approach as an example of using past search behaviors for positive relevance feedback due to its popularity[2]. Shen et al. [8] proposed four query model estimation methods. In our experiments, we adopt the "FixInt" method because both Shen et al. and we found that it outperforms the other methods.

Let q_k be the kth query in a search session, FixInt estimates query language model θ_k as Eq(1): $P(w|q_k)$ is the current query's MLE model; $P(w|H_c)$ and $P(w|H_q)$ are, respectively, relevance feedback models estimated based on click-through documents and previous queries. Eq(2) and Eq(3) show details of H_c and H_q: C_i is the concatenation of all clicked documents' summaries returned by q_i; $P(w|q_i)$ is the ith query's MLE model. Parameter α is the weight of the current query in the FixInt query language model.

$$P(w|\theta_k) = \alpha P(w|q_k) + (1-\alpha)\left[\beta P(w|H_c) + (1-\beta) P(w|H_q)\right] \quad (1)$$

$$P(w|H_c) = \frac{1}{k-1}\sum_{i=1}^{k-1} P(w|C_i) \quad (2)$$

$$P(w|H_q) = \frac{1}{k-1}\sum_{i=1}^{k-1} P(w|q_i) \quad (3)$$

2.2 Demoting Repeated Results

We assume that the usefulness of a result for the user degrades each time viewing the result or its snippet. Therefore, repeatedly occurred results within a search session should be demoted in a rank list. We demote the rankings of the repeatedly results by $P(d|s)$, which can be explained as: the probability that the user will still be interested in examining the document d provided the session search history s, which typically includes past search queries and results. We explained this approach in details in our previous studies [1, 2].

We assume the following user behaviors:

- The user examines results by sequence from top to bottom. The user will always examine the first result in a result page. After examined each result, the user has probability p to continue examining the next one, and probability $1 - p$ to stop (either to reformulate a new query for search or to terminate the current session). This browsing model is similar to the one adopted in rank-biased precision [6].
- Each time the user examined a result, it has probability β that the result will lose its attractiveness to the user in the rest of the search session.

According to these assumptions, as in Eq(4), a document d can keep its attractiveness if and only if it did not lose attractiveness in any of the previous searches. In Eq(4): $R^{(i)}$ refers to the results for the ith query in the session (assuming q is the nth query); $P_{\text{examine}}(d|R^{(i)})$ is the probability that d will be examined when the user browses results $R^{(i)}$, as calculated in Eq(5); $rank(d, i)$ is the rank of d in $R^{(i)}$. According to this model, a document will be demoted to a larger magnitude if it was retrieved by many of the previous queries and/or it was ranked at top positions.

$$P(d|s) = 1 - \prod_{i=1}^{n-1}\left(1 - \beta \cdot P_{\text{examine}}(d|R^{(i)})\right) \quad (4)$$

$$P_{\text{examine}}(d|R^{(i)}) = \begin{cases} p^{rank(d,i)-1} & d \in R^{(i)} \\ 0 & d \notin R^{(i)} \end{cases} \quad (5)$$

2.3 Alternative Evaluation Approach

We use the TREC session track 2012 datasets for evaluation. The dataset includes static search session logs and whole-session level relevance judgments. A static search session is the search history of a real user in an interactive search system, including the users' search queries, click-through, and other information. For each static search session, whole-session level relevance judgments are provided in the datasets: annotators judged documents regarding whether or not they are relevant to the topic or task underlying the search session (instead of an individual query).

The past TREC session tracks evaluated participant systems based on nDCG@10 of the last queries in each session. In comparison, we adopted the following experiment procedure to study the whole-session search performance. Let $\{q_1, q_2, \ldots, q_n\}$ be a static search session in the dataset. We iteratively produce results for $q_1, q_2, \ldots, q_n$ using FixInt. For each q_i, we use the past queries and click-through (if any) in the static search session log as positive relevance feedback information to produce search results. For q_1, FixInt downgrades to query likelihood model.

After generated results for each query in a static session, we calculate the following measures:

(1) **nDCG@10** (macro-average). Let $\{S_1, S_2, \ldots, S_m\}$ be m static search sessions in a dataset, and $\{R_{i1}, R_{i2}, \ldots, R_{in}\}$ be the results of the n queries in session S_i. We calculate the macro-average nDCG@10 of the dataset (referred to as nDCG@10) as follows:

$$\frac{1}{m}\cdot\sum_{i=1}^{m}\left(\frac{1}{n-1}\cdot\sum_{j=2}^{n} nDCG@10(R_{ij})\right)$$

Note that we do not count the first query of each session because for the first query there is no search history information available (and here we mainly hope to examine the effect of the past search history to search performance).

(2) **nsDCG@10** (normalized session DCG). For a static search session, nsDCG@10 concatenates the top 10 results of each query and evaluates the performance of the concatenated list of results. Please refer to [4] for details. The same parameters have been adopted in this study.

(3) **Instance recall (instRec)**. This is a variant of a major metric adopted in the early TREC interactive tracks [7]. In previous TREC interactive tracks, annotators identified relevant instances of each topic and marked up the occurrences of relevant instances in documents. The metric "instance recall" was originally calculated as the proportion of relevant instances covered by the search results over all the identified instances. Here we calculate a similar measure by considering each single relevant document as a unique instance.

Let $\{D_i\}$ be the top 10 results of q_i, and D_R be the set of judged relevant documents. We concatenate the top 10 results of each query in the session as a whole set of retrieved documents (D_F).

[2] By the time of finalizing our report, Shen et al.'s article [8] has been cited 329 times according to Google Scholar.

Then, we calculate instance recall (instRec) of the session as the proportion of D_R covered by D_F.

$$D_F = \bigcup_{i=1}^{n} \{D_i\} \qquad \text{instRec} = \frac{|D_F \cap D_R|}{|D_R|}$$

(4) **Jaccard similarity**. For a static session, we calculate for each unique pair of queries the Jaccard similarity of the pair of queries' top 10 results. Then, we calculate the macro-average value for each unique pair of queries across all search sessions. Although jaccard similarity is not a metric of search performance, it can help us analyze the novelty of search results.

2.4 Results

We separately examine the effects of past queries and clicked results to whole-session search performance. Figure 1 and Figure 2 show the whole-session search performance of FixInt model using sorely past queries or clicked results with different weights α. Figure 3 and Figure 4 further shows the whole-session search performance of FixInt using past queries or click-through results after demotion of repeated results.

Our results indicate that:

(1) As we suspected in [2], past queries can lead to serious decline of search novelty by making the results of the current query similar to previous queries' results. As shown in Figure 1, in all types of tasks, the average Jaccard similarity of top 10 results can be increased from around 0.3 (α = 1.0, using sorely the user query) to around 0.8 (α = 0, using sorely the past queries). Whenever we increase the weight of past queries, there will be increase of the jaccard similarity.

When nDCG@10 reaches the peak value, although we achieve 10% – 20% increase in nDCG@10, there is also 0.1 – 0.2 increase of jaccard similarity. In addition, instance recall dropped from 0.088 (α = 1.0) to 0.082 (α = 0.5) in the 2012 dataset (counting all types of tasks). This makes it difficult to assess whether there is a true improvement of search performance, because the increase of nDCG@10 may come from previously retrieved relevant results.

Overall we found that past queries have no apparent effect of improving whole-session search performance such as instance recall. Average nDCG@10 of queries does not consider the novelty of search results and therefore cannot disclose the drop of novelty in FixInt. As shown in Figure 1, instRec can at most be increased from 0.0881 (α = 1.0) to 0.0896 (α = 0.8). We also did not observe that FixInt performed differently in any type of tasks.

(2) In comparison, our results suggest that click-through is a valuable relevance feedback information for improving search performance without sacrificing novelty of results. As shown in Figure 2, using click-through documents, the jaccard similarity of results will at most increase by about 0.1 (comparing to the increase from about 0.3 to 0.8 when using past queries).

In all types of tasks, click-through can increase nDCG@10 by 10% – 20%. In addition, instRec can also be increased by 10%, from 0.088 (α = 1.0) to 0.101 (α = 0.5) (counting all types of tasks). When instRec reaches the peak value, there are also about 10% increase of nDCG@10, indicating that click-through may improve the ranking of relevant documents without sacrificing novelty of results, which result in performance improvement all over the search session (as indicated from instRec).

(3) Table 1 shows the correlation (Pearson's r) of various metrics on different parameter values of α and β. Results indicate that nDCG@10 & nsDCG@10 have slight negative correlation with instRec.

Table 1. Pearson's correlation of metrics on different parameter values of α and β.

	TREC 2011		TREC 2012	
	nDCG@10	**instRec**	**nDCG@10**	**instRec**
nDCG@10	1.000	-0.235	1.000	0.245
nsDCG@10	0.985	-0.244	0.994	0.204
instRec	-0.235	1.000	0.245	1.000
avgJaccard	0.413	-0.957	0.180	-0.890

(4) Our approach of demoting repeated results leads to slight decrease of nDCG@10 but apparent increase of instRec. After selecting appropriate parameters, it can achieve improved search performance balancing nDCG@10 and novelty of results. For example, as shown in Figure 3 (FixInt with past queries), when setting both p and β to 0.5, we can increase nDCG@10 from 0.252 (α = 0 and do not demote repeated results) to 0.272 (α = 0.8, p = 0.5, β = 0.5) and at the same time increase instRec from 0.088 to 0.106. This indicates that past queries should be combined with proper approaches demoting repeated results in order to balance search performance and novelty of results.

3. SUBMITTED RUNS

Based on our observations, we submitted three groups of runs:

(1) FixInt28: a FixInt run optimizing macro-average nDCG@10 (α = 0.2, β = 0.8). FixInt28N further applied the ranking discount method we proposed last year to FixInt28.

(2) FixInt58: a FixInt run optimizing instance recall (α = 0.5, β = 0.8). FixInt58N further applied the ranking discount method we proposed last year to FixInt58.

(3) KM1 and KM1N: new retrieval models aiming at whole-session relevance. Unfortunately, this new approach did not work well. Due to the lack of qrels at the time of finalizing our report, we do not examine details of this approach.

4. CONCLUSIONS

We evaluated the effects of using past queries and click-through as positive relevance feedback information on whole-session search performance. Our results indicate that it is risky to utilize past queries because we may easily sacrifice novelty of results to enhance search performance. In comparison, click-through seems to be a more valuable resource to balance search performance and novelty of results. However, after demoted repeated results in a search session, we can balance search performance and novelty of results using either past queries or click-through effectively. This also indicates that it is very necessary to demote repeated results in a search session.

5. REFERENCES

[1] Jiang, J. et al. 2012. Contextual evaluation of query reformulations in a search session by user simulation. *Proceedings of the 21st ACM international conference on Information and knowledge management (CIKM '12)* (New York, New York, USA, Oct. 2012), 2635.

[2] Jiang, J. et al. 2012. On Duplicate Results in a Search Session. *Proceedings of the 21st Text REtrieval Conference, (TREC 2012)* (2012).

[3] Jiang, J. et al. 2011. Pitt at TREC 2011 session track. *Proceedings of the 20th Text REtrieval Conference, (TREC 2011)* (2011).

[4] Kanoulas, E. et al. 2010. Session track overview. *The 19th Text REtrieval Conference Notebook Proceedings (TREC 2010)* (2010).

[5] Lafferty, J. and Zhai, C. 2001. Document language models, query models, and risk minimization for information retrieval. *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval* (New York, NY, USA, 2001), 111–119.

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

[7] Over, P. 2001. The TREC interactive track: an annotated bibliography. *Information Processing & Management*. 37, 3 (2001), 369–381.

[8] Shen, X. et al. 2005. Context-sensitive information retrieval using implicit feedback. *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '05* (New York, New York, USA, Aug. 2005), 43.

[9] Zhai, C. and Lafferty, J. 2001. Model-based feedback in the language modeling approach to information retrieval. *Proceedings of the tenth international conference on Information and knowledge management* (2001), 403–410.

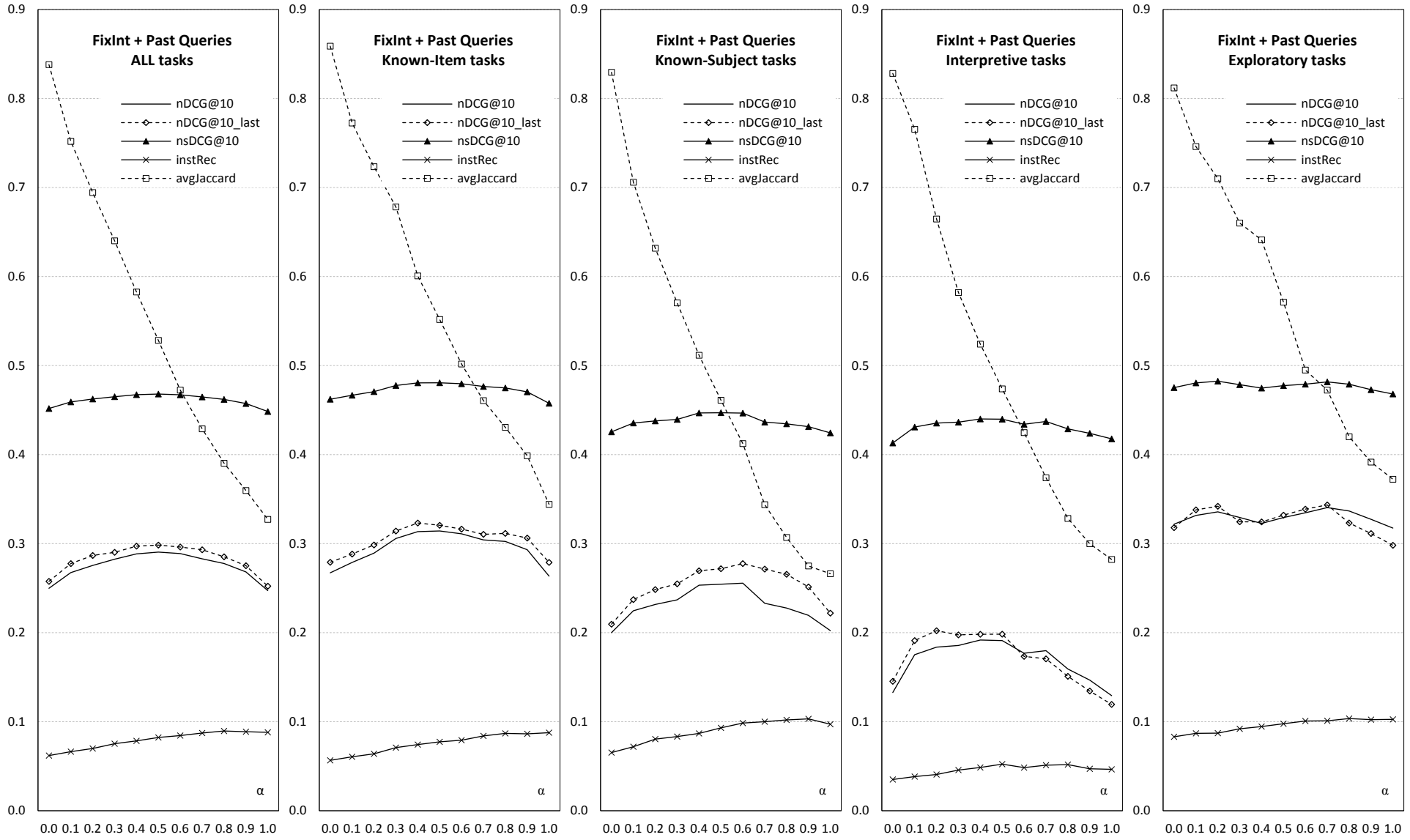

Figure 1. The weight of past queries as positive relevance feedback information in FixInt and the corresponding whole-session search performance (the greater the value of α, the smaller the weight of past queries in FixInt; α = 1.0 means only using user query for ranking, and α = 0 means sorely using past queries).

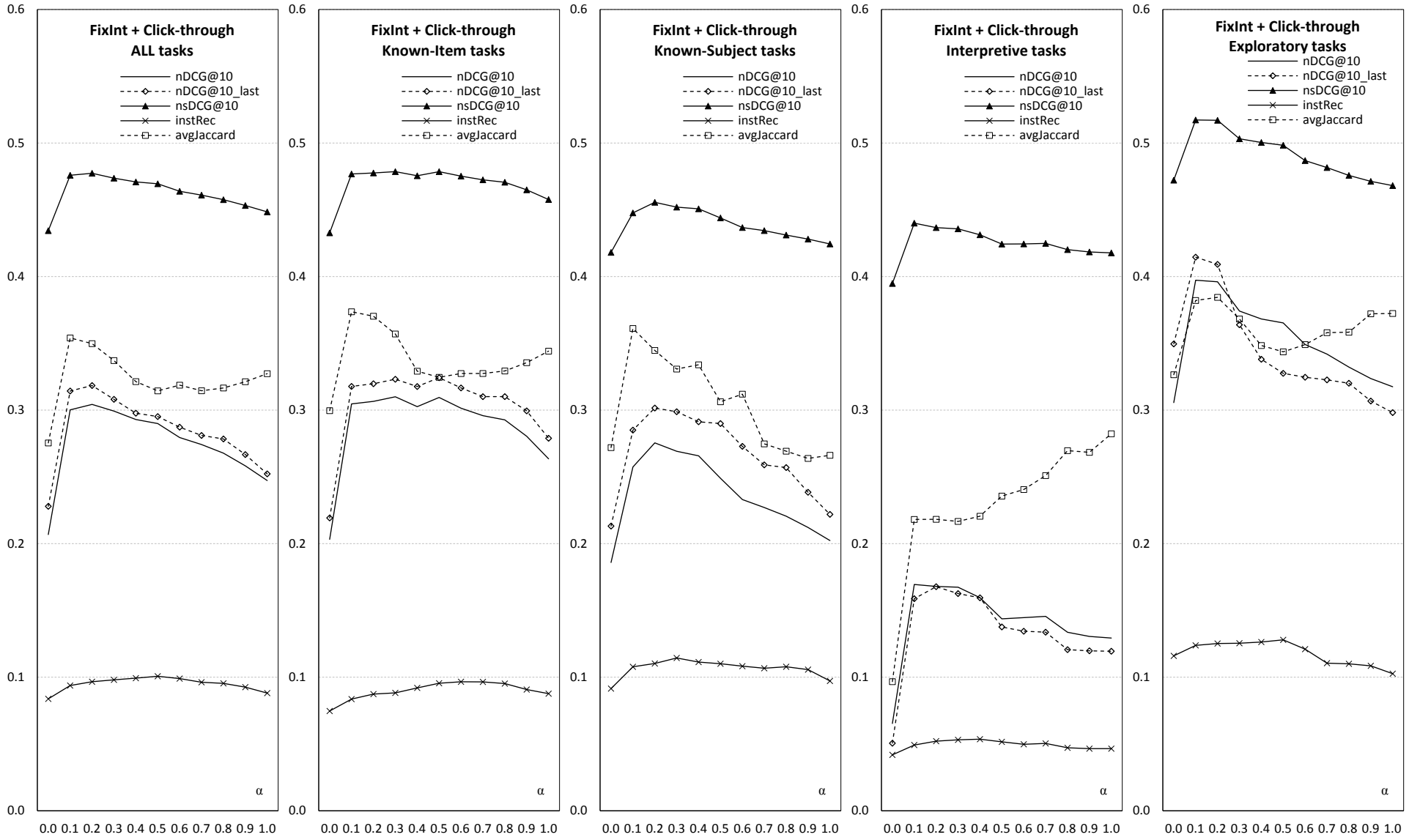

Figure 2. The weight of click-through as positive relevance feedback information in FixInt and the corresponding whole-session search performance (the greater the value of α, the smaller the weight of click-through in FixInt; α = 1.0 means only using user query for ranking, and α = 0 means sorely using click-through).

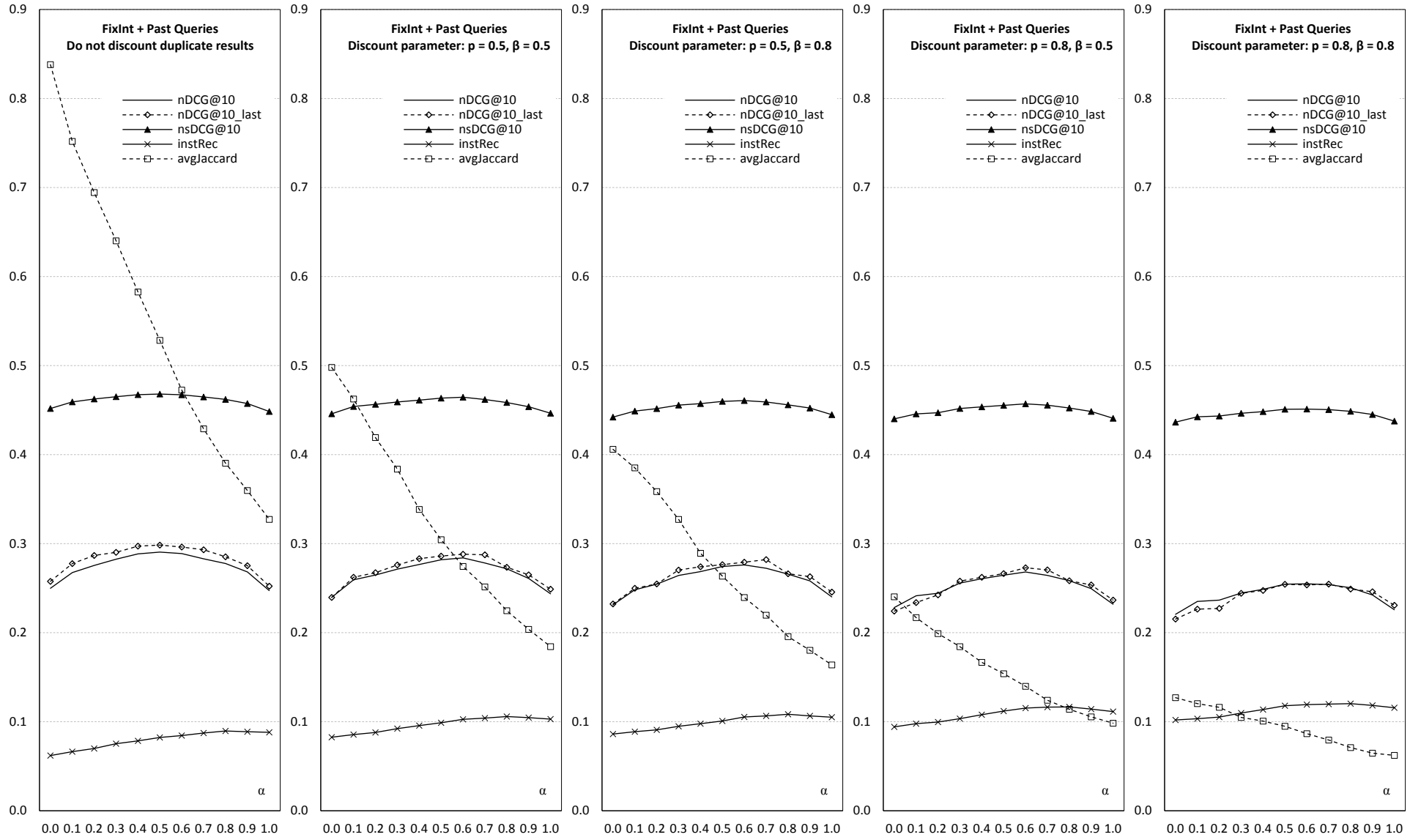

Figure 3. The weight of past queries as positive relevance feedback information in FixInt (after demoting repeated results) and the corresponding whole-session search performance (the greater the value of α, the smaller the weight of past queries in FixInt; α = 1.0 means only using user query for ranking, and α = 0 means sorely using past queries).

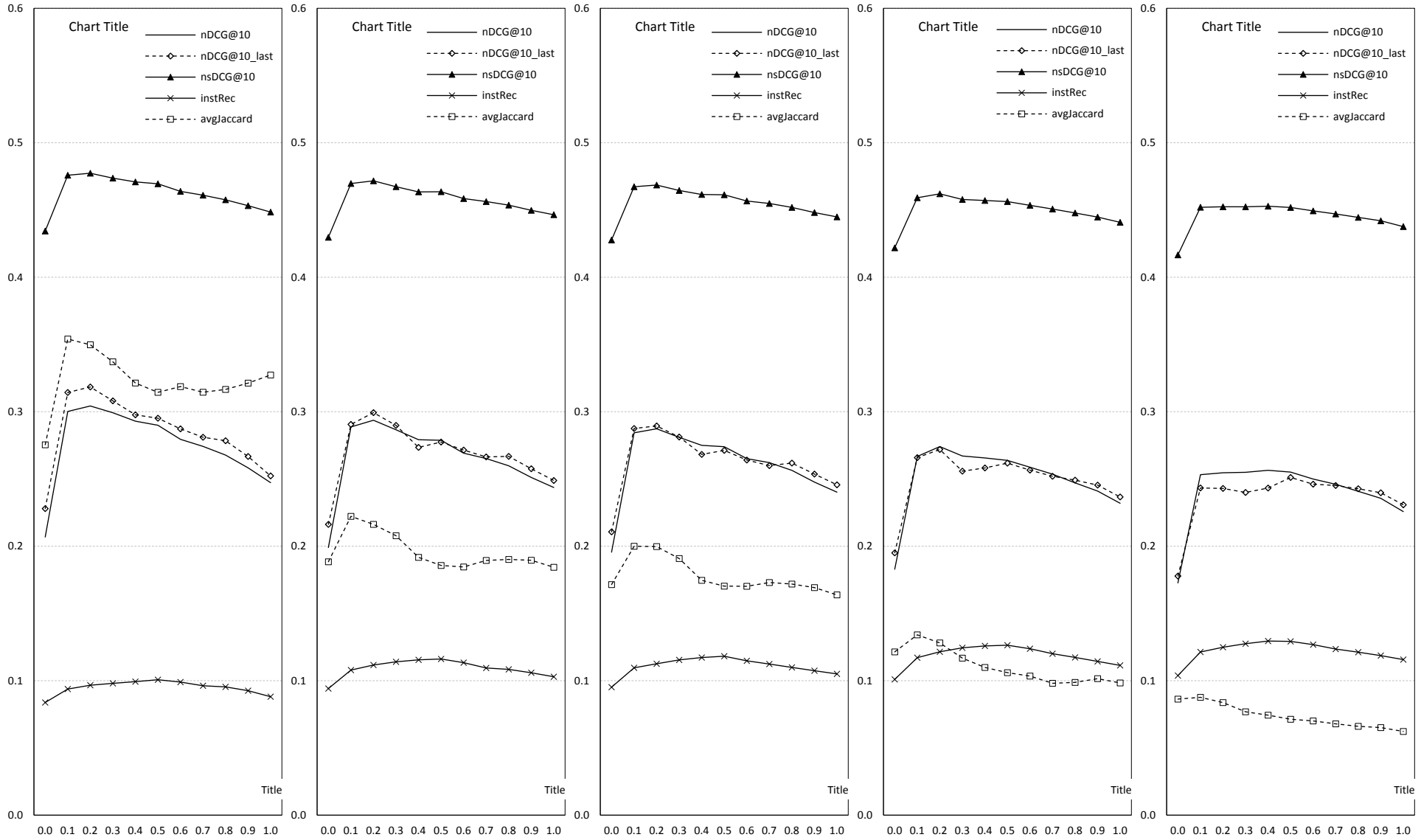

Figure 4. The weight of click-through as positive relevance feedback information in FixInt (after demoting repeated results) and the corresponding whole-session search performance (the greater the value of α, the smaller the weight of click-through in FixInt; α = 1.0 means only using user query for ranking, and α = 0 means sorely using click-through).