Ranking Retrieval Systems without Relevance Assessments – Revisited

Tetsuya Sakai
Microsoft Research Asia, PRC
tetsuyasakai@acm.org

Chin-Yew Lin
Microsoft Research Asia, PRC
cyl@microsoft.com

ABSTRACT

We re-examine the problem of ranking retrieval systems without relevance assessments in the context of collaborative evaluation forums such as TREC and NTCIR. The problem was first tackled by Soboroff, Nicholas and Cahan in 2001, using data from TRECs 3-8 [16]. Our long-term goal is to semi-automate repeated evaluation of search engines; our short-term goal is to provide NTCIR participants with a "system ranking forecast" prior to conducting manual relevance assessments, thereby reducing researchers' idle time and accelerating research. Our extensive experiments using graded-relevance test collections from TREC and NTCIR compare several existing methods for ranking systems without relevance assessments. We show that (a) The simplest method of forming "pseudo-qrels" based on how many systems returned each pooled document performs as well as any other existing method; and that (b) the NTCIR system rankings tend to be easier to predict than the TREC robust track system rankings, and moreover, the NTCIR pseudo-qrels yield fewer false alarms than the TREC pseudo-qrels do in statistical significance testing. These differences between TREC and NTCIR may be because TREC sorts pooled documents by document IDs before relevance assessments, while NTCIR sorts them primarily by the number of systems that returned the document. However, we show that, even for the TREC robust data, documents returned by many systems are indeed more likely to be relevant than those returned by fewer systems.

Categories and Subject Descriptors

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

General Terms

Experimentation

Keywords

test collections, pooling, relevance judgments, evaluation

1. INTRODUCTION

Search engine companies routinely conduct manual relevance assessments of web pages for thousands of queries for evaluation and improvement. Since manual assessments are expensive and time-consuming, the idea of evaluating systems without relevance assessments is attractive.

In their very first attempt at ranking retrieval systems without relevance assessments, Soboroff, Nicholas and Cahan [16] remarked that "such a methodology would not be useful in an environment such as TREC, where there is already a commitment to conducting relevance assessments and building test collections in the Cranfield tradition." In fact, our main motivation for tackling this difficult problem is to accelerate research at evaluation forums such as TREC and NTCIR.

Take the NTCIR-6 crosslingual task, for example. The run (i.e. system output) submission deadline was August 1, 2006, and the evaluation results were released on November 29, 2006 [8]. The four months were necessary for the organisers to form "qrels" (i.e. relevance assessment results) for multiple document languages and to rank submitted runs, so the participants had no choice but to wait. The "idle time" was reduced to two months at the NTCIR-7 ACLIA IR4QA task (another "ad hoc" IR task), but this resulted in tentative qrels [12], and their updated versions were released six months after NTCIR-7 [13].

If the organisers can release a "system ranking forecast" with reasonable accuracy right after the run submission deadline, in which at least some good and bad systems are identified, this may help researchers to conduct some focussed experiments while waiting for the "true" system ranking to arrive. These experiments may provide some preliminary lessons to participants[1].

This line of research is complementary to the existing work on constructing test collections economically (e.g. [5, 15]) and those on evaluation with incomplete relevance assessments (e.g. [11]).

The main contributions of this paper are as follows. Using graded-relevance test collections from both TREC and NTCIR, we implement and compare most of the existing methods for ranking systems without relevance assessments, and show that (a) The simplest method of forming "pseudo-qrels" based on how many systems returned each pooled document performs as well as any other existing method; and that (b) the NTCIR system rankings tend to be easier to predict than the TREC robust track system rankings, and moreover, the NTCIR pseudo-qrels yield fewer false alarms than the TREC pseudo-qrels do in statistical significance

EVIA 2010 June 15, Tokyo, Japan.

[1]At NTCIR, participants are required to submit a brief system description together with each run. The system descriptions could be shared among the participants together with the ranking forecast.

testing. These differences between TREC and NTCIR may be because TREC sorts pooled documents by document IDs before relevance assessments [19], while NTCIR sorts them primarily by the number of systems that returned the document[2]. However, we show that, even for the TREC robust data, documents returned by many systems are indeed more likely to be relevant than those returned by fewer systems.

2. RELATED WORK

Soboroff, Nicholas and Cahan [16] were the first to tackle the problem of ranking runs without relevance assessments given pooled documents. Their idea was to replace manual relevance assessments by random sampling from the pool, as it is known that different relevance assessment sets can yield similar system rankings [18]. Their method assumed knowledge of the mean and standard deviation of the number of truly relevant documents, which is unavailable in practice. Hence, we implemented a variant of this method, following a previous study [22]. Furthermore, since Soboroff *et al.* found that retaining duplicate documents in the pool (i.e. documents returned by multiple runs) for sampling improves system ranking accuracy, we follow this particular method.

Wu and Crestani [22] proposed several methods as alternatives to "Soboroff's method" and actually compared them in terms of system ranking accuracy. We do not include their methods in our experiments because (a) they reported that Soboroff's method performed better; (b) their best methods rely either on the actual document scores[3] or an arbitrary parameter (K) for linear discounting of documents based on document ranks; (c) their basic idea of comparing documents across runs is also explored by other approaches examined in this paper.

Aslam and Savell [3] introduced a very simple method for ranking runs without relevance assessments, based on how runs resemble one another, and showed that it correlates very highly with Soboroff's method. We re-examine this method as well.

Nuray and Can [10] also proposed methods for ranking runs without relevance assessments. Their supposedly best method, called "condorcet bias," consists of two parts: (i) select top 50% of the entire set of runs based on *bias* as defined by Mowshowitz and Kawaguchi [9], which is supposed to quantify how different each system is from the "norm"; (ii) rank the documents returned by the selected runs by the condorcet measure and treat the top s% as relevant. When the run selection process (i) is omitted, the method is called "condorcet normal." We implement both of these methods. Nuray and Can [10] did *not* directly compare their methods with previously proposed ones in their experiments.

[2]Note that NTCIR does *not* pool documents based on the number of systems that returned each document: just like TREC, NTCIR takes the top ranked documents from each run to form a pool. The only difference is the *order* of documents presented to the relevance assessors. We would like to stress this at this point, as NTCIR has received multiple criticisms which incorrectly claim that NTCIR pools documents based on popularity and therefore the NTCIR test collections are biased. To our knowledge, whether the *order* of relevance assessments really biases them and affect evaluation results is an open question [12].

[3]If we want to evaluate commercial search engines without relevance assessments, for example, we may not have access to the document scores.

Table 1: Data used in previous work (in chronological order).

year	authors	data
2001	Soboroff *et al.* [16]	TREC-3,5,6,7,8
2003	Wu/Crestani [22]	ditto
2003	Aslam/Savell [3]	ditto
2006	Nuray/Can [10]	TREC-3,5,6,7
2007	Spoerri [17]	TREC-3,6,7,8
2009	Sakai *et al.* [13]	NTCIR IR4QA-CS,CT,JA

Spoerri [17] proposed another method which also examines the "uniqueness" of systems. His method is for ranking participating *teams* rather than the entire set of runs: "it is critical that only one run by each participating system is included" [17]. This is because uniqueness, as measured by the number of documents returned by one particular run only, cancels out when two similar runs from the same team are considered. We therefore evaluate his method in our *team* ranking experiments, which we conduct in addition to our *run* ranking experiments. Again, Spoerri [17] did *not* compare his method with existing alternatives.

Sakai *et al.* [13] proposed a very simple method for ranking runs without relevance assessments in the context of the NTCIR ACLIA IR4QA task [12]. They sorted the pooled documents first by the number of runs that returned the document, then by the sum of the ranks of that document within those runs. In fact, at IR4QA, the manual relevance assessments themselves were done after applying this very sort. They reported that the correlations between their pseudo-qrels and their "true" qrels were very high, but questions remained: does the method work with other data? How does it compare to other automatic ranking methods?

As Table 1 shows, all of the abovementioned studies used TREC data with binary relevance assessments, except for that by Sakai *et al.* [13] who used Simplified Chinese (CS), Traditional Chinese (CT) and Japanese (JA) data with graded relevance assessments. In contrast, we use graded-relevance data from both of the evaluation forums (which enables us to use graded-relevance retrieval effectiveness metrics), *and* examine a broad choice of automatic system ranking methods mentioned above.

3. SYSTEM RANKING ALGORITHMS

This section describes our implementations of existing algorithms for ranking retrieval systems without relevance assessments. We use formal and precise descriptions as much as possible to enhance reproducibility. We use documents ranked at or above 30 within each run[4], and ignore document scores. While both "Aslam's method" and "Spoerri's method" produce a single system ranking without producing pseudo-qrels, all other methods yield pseudo-qrels that can be used just like "true" qrels for computing different evaluation metrics: "Soboroff's method" in fact produces many pseudo-qrels.

Some of these methods require a few parameters, which we shall discuss in Section 4.

3.1 Soboroff et al. ("Soboroff's Method")

Our version of this method randomly samples 10% of the documents in the depth-30 pool (containing duplicates) for

[4]We used the pool depth of 30 rather than 100, because (a) Soboroff *et al.* reported that shallow pools may work better for this task; and (b) shallow pools reduce computational cost: see Section 3.4.

each topic and treats them as relevant [16, 22]. This is done 10 times to produce 10 different pseudo-qrels files. (Soboroff et al. [16] sampled 50 times, but the variance across our trials was actually very small.) Hence, for each, run, we obtained 10 different performance values, and we averaged them to produce a single system ranking. (We also tried evaluating individual system rankings and averaging the rank correlations [16], but the final outcome was very similar.)

3.2 Aslam/Savell ("Aslam's Method")

Let n be the total number of submitted runs and let $Ret_{i,t}$ denote the set of documents returned at or above rank 30 by run i for topic $t(\in T)$. Rank runs by how each run (as a document set) resembles all other runs [3]: that is, by

$$\frac{1}{|T|}\sum_t \frac{1}{n-1} \sum_{j\neq i,\ Ret_{j,t}\neq\phi} \frac{|Ret_{i,t}\cap Ret_{j,t}|}{|Ret_{i,t}\cup Ret_{j,t}|} \ .$$

3.3 Spoerri ("Spoerri's Method")

Select one run from each participating team. In our experiments, we simply sort the runs by their file names and take the first run from each team, since this is more reproducible than random selection. (In the original paper [17], the "best" run from each team was chosen. In this sense, Spoerri's original experiments actually relied on true qrels.)

If there are K teams, K runs are thus selected. From this set of runs, select five runs at random (without replacement) for K trials, so that each run is selected in exactly five trials. (This is in essence to generate an $K\times K$ bit matrix in which each row/column adds up to 5.) Let $S(k)$ denote the set of runs sampled in the k-th trial, where $|S(k)|=5$ for $1\leq k\leq K$. Rank runs by

$$-\frac{1}{5}\sum_{k,\ i\in S(k)} \frac{1}{|T|}\sum_t \frac{|Ret_{i,t}-\cup_{j\in S(k),j\neq i}Ret_{j,t}|}{1+|Ret_{i,t}|}$$

which reflects the proportion of documents returned by i and no other run for each trial[5]. (Spoerri [17] also proposed a variant of this method that relies on the set of documents returned by all five systems in each trial. We evaluated this variant too, but the outcome was similar.)

3.4 Nuray/Can ("Nuray's Methods")

As mentioned earlier, this method consists of two parts: run selection by bias, followed by document selection by condorcet. We tried both "condorcet bias" and "condorcet normal," i.e. with and without run selection [10].

3.4.1 Run Selection by Bias

Let $P_t=\cup_i Ret_{i,t}$, i.e. the depth-30 pool for topic t. Let $P=\cup_t P_t$. For each run i and for each topic t, compute the "response vector" $resp_{i,t} =< r_1, r_2, \ldots, r_{|P|} >$, where $r_l = 30/m$ if run i for topic t contains the l-th document of P at rank $m(\leq 30)$, and $r_l=0$ otherwise. Then for each run i, let $Resp_i =< R_1, R_2, \ldots, R_{|P|} >$ where $R_l=\sum_t r_l$. Let $RESP =< S_1, S_2, \ldots, S_{|P|} >$ where $S_l=\sum_i R_l$. Sort the runs by how different each run is from the "norm" [9]:

$$BIAS(i)=1-\cos(Resp_i, RESP)\ .$$

[5]The minus sign is for obtaining positive correlations with retrieval performance. The "1" in the denominator avoids division by zero: the same applies to our implementation of condorcet described in Section 3.4.2.

Take the top 50% most "biased" runs.

3.4.2 Document Selection by condorcet

For every document pair $d(\in P_t)$ and $d'(\in P_t)$ (if run selection described above is applied, then define the depth-30 pool P_t over the *selected* runs instead of *all* runs.), compute $win_t(d,d')$, the number of runs in which d is ranked above d', and $lose_t(d,d')$, the number of runs in which d is ranked below d'. (If d is ranked within top 30 and d' is either below rank 30 or not retrieved, then d "wins".) Sort $d(\in P_t)$ by

$$condorcet(d)=\sum_{d'} win_t(d,d')+\frac{1}{1+\sum_{d'} lose_t(d,d')}\ .$$

That is, sort by the total number of wins (the larger the better), and then for each tie, sort by the total number of losses (the smaller the better). Finally, treat the top 30% documents as relevant. This is a rather computationally expensive method, since the ranks must be compared for all pairs of pooled documents.

3.5 Sakai et al. ("Sakai's Method")

For each document $d(\in P_t)$, let $runs_t(d)=\{i|d\in Ret_{i,t}\}$ and let $rank_{i,t}(d)$ denote the rank of d for topic t in run $i(\in runs_t(d))$. Sort the documents in P_t by

$$|runs_t(d)|+\frac{1}{\sum_{i\in runs_t(d)} rank_{i,t}(d)}\ .$$

That is, sort by the number of runs (the larger the better), and then for each tie, sort by the sum of ranks (the smaller the better). Treat the top 30% as relevant. (The original method [13] treated the top 100 documents as relevant. We also tried this method, but the results were very similar.)

We also tried a simple variant of Sakai's method that relies only on $|runs_t(d)|$ for selecting pseudo-relevant documents. We call this variant "nruns."

3.6 On condorcet vs. Sakai's Method

Here, we show that the primary sort key of condorcet and the primary sort key of Sakai's method (i.e. "nruns") are in effect very similar. Recall that condorcet is a relatively expensive method. Sakai's method is not.

Suppose d is ranked above d' for every $d'\in P_t$ $(d'\neq d)$ and for every run $i(\in runs_t(d))$. Then, by definition, $\sum_{d'} win(d,d') = |runs_t(d)|*(|P_t|-1)$. That is, the primary sort key of condorcet is proportional to that of Sakai's method. Now, d is in practice ranked "above" d' for many d''s, because the size of the depth-30 pool $|P_t|$ is much larger than $|Ret_{i,t}|(\leq 30)$: usually hundreds [12].

On the other hand, these two statistics are in theory not equivalent. For example, suppose d_1 is retrieved by exactly $n'(>1)$ runs, all at rank 30, while d_2 is retrieved by exactly one run at rank 1, so that Sakai's method prefers d_1 to d_2. It is easy to show that condorcet's first key prefers d_2 to d_1 if $|P_t|<(30n'-1)/(n'-1)$.

4. DATA

The top half of Table 2 provides some statistics on the TREC and NTCIR "ad hoc" IR data sets that we used. "ROBUST03" and "ROBUST04" are from the TREC 2003 and 2004 Robust tracks, and we used the "new" topics [20, 21]; "CLIR6-JA" and "CLIR6-CT" are from the Japanese

Table 2: Statistics on test collections, runs and pseudo-qrels used in our experiments.

	ROBUST03	ROBUST04	CLIR6-JA	CLIR6-CT	IR4QA-CS
#all runs (teams)	78 (16)	110 (14)	74 (12)	46 (11)	40 (9)
#topics	50	49	50	50	97
#documents	approx. 528,000		858,400	901,446	545,162
pool depth of "true" qrels	125	100	100	100	100
$\|P\|$ (See Section 3.4.1)	11,835	15,542	19,280	18,746	18,361
#total relevant/topic	33.2	41.2	95.3	88.1	169.8
#$L3/L2/L1$-relevant/topic	8.1/-/25.0	12.5/-/28.8	2.5/61.1/31.7	21.6/30.4/36.1	-/108.2/61.6
#pseudo-rel. (soboroff10%)/topic	95.2	120.9	113.9	89.4	66.4
#pseudo-rel. (nuray30%, sakai30%, nruns30%)/topic	78.7	104.6	133.2	123.2	61.8
#pseudo-rel. (BIAS+nuray30%)/topic	66.2	93.8	110.3	103.3	41.8
#pseudo-rel. (TEAM+soboroff10%)/topic	34.3	32.7	29.3	27.2	21.4
#pseudo-rel. (TEAM+nruns30%)/topic	43.5	44.4	52.4	49.4	39.3

and Traditional Chinese subtasks of the NTCIR-6 Crosslingual task [8]; and "IR4QA-CS" is from the Simplified Chinese subtask of the NTCIR-7 ACLIA IR4QA task [12]. Following a practice at NTCIR [12], highly relevant, relevant and partially relevant documents are referred to as $L3$-, $L2$- and $L1$-relevant documents, respectively. Following a previous study that used both TREC and NTCIR data [11], the (regular) relevant documents of "ROBUST03" and "ROBUST04" were treated as $L1$-relevant rather than $L2$-relevant.

The bottom half of Table 2 shows the number of pseudo-relevant documents per topic for some of the methods we examined. As was described in Section 3.1, we used 10% samples from the document pool (containing duplicates) for Soboroff's method because we did not want to assume knowledge of the true number of relevant documents per topic. As for Nurray's, Sakai's and the "nruns" methods, this paper reports on results that used the top 30% documents of the sorted list of pooled documents for forming pseudo-qrels. This percentage was chosen so that the number of pseudo-relevant documents per topic based on the four methods are comparable: note that unlike Soboroff's method, the other three methods use document pools without duplicates[6].

"BIAS+nuray30%" and "nuray30%" represent Nuray's method with and without run selection, respectively, and we use similar notations for other methods. Note that "BIAS+nuray30%" yields fewer pseudo-relevant documents than "nuray30%", as the number of candidate documents is reduced through run selection. "TEAM+soboroff10%" represents Soboroff's method applied after selecting one run per team as was described in Section 3.3 for the task of ranking *teams* rather than runs: this is also a kind of run selection and therefore reduces the size of pseudo-qrels. Using a higher percentage value with the BIAS and the TEAM methods did not improve their ranking prediction accuracy, however.

We also conducted some "oracle" pseudo-qrels experiments, in which we ensured that the number of pseudo-relevant documents is equal to that of true relevant documents for every topic. However, we will omit these results as this did not necessarily improve the ranking prediction accuracy. This is in agreement with an observation by Soboroff *et al.*: "the ranking of systems is not nearly as affected by variation in the number of relevant documents as it is by which specific documents are selected" [16].

[6]We tried a few other percentage values for all four methods, and found that the methods are not very sensitive to the parameter choice. Moreover, it is more useful to observe general trends across multiple and diverse data sets than tweaking parameter values that can easily lead to overfitting. See also the above discussion on the "oracle" experiments.

5. METRICS

For evaluating system performance, we adopt the three official metrics used at NTCIR ACLIA IR4QA: Average Precision (AP), Q-measure (Q) and a version of nDCG [12]. Q and nDCG, which can handle graded relevance, used the gain values of 3/2/1 for $L3/L2/L1$-relevant documents, respectively. Due to lack of space, our tables show results with AP and nDCG only; whereas, our graphs show results with Q, as it behaves like a summary of AP and nDCG [12].

For comparing two system rankings, we use Kendall's τ rank correlation and its "top-heavy" variant called τ_{ap} [12, 23]. While we acknowledge some problems with τ for the purpose of IR evaluation [4], we believe that it is still useful as a simple single-value summary. Moreover, using it together with τ_{ap} provides an additional insight: for example, if τ_{ap} is much lower than τ, this means that the top ranks are quite inaccurate [23]. We note, however, that the raw system ranking graphs deserve more attention than rank correlation values.

6. EXPERIMENTS

6.1 Ranking Runs

Table 3 compares the predicted system rankings with the "true" system rankings in terms of τ and τ_{ap}. Note that small differences in these correlation values are not meaningful. It can be observed that:

(1) BIAS+nuray30%, which uses only half of the available runs through computation of "bias" (see Section 3.4.1), is not effective.

(2) All of the other methods show comparable performances, although Aslam's method is less robust in that it fails for the IR4QA-CS data.

Observation (1) challenges Nuray and Can's claim that run selection by "bias" is effective [10]. However, we note that bias hurt accuracy even in *their* experiments with the TREC-7 data. As Section 3.4.1 showed, bias aggregates the distribution of retrieved documents across topics and forms a sparse, high-dimensional vector for each system (the exact dimension is shown in the "$|P|$" row in Table 2). The idea is to rely on systems with a novel document distribution for selecting pseudo-relevant documents, but this did not work in our experiments. If system "novelty" is useful at all for predicting system ranking, a better approach may be to define a measure of novelty *per topic*. Another possible approach to "purifying" the clues for predicting system ranking would be to perform some kind of *topic set reduction* [6], thereby removing topics with high uncertainty.

Table 3: τ and τ_{ap} rank correlations: "true" run ranking vs predicted *run* ranking. For each metric, the three highest values in each column are shown in bold (a tie is counted as one).

		ROBUST03	ROBUST04	CLIR6-JA	CLIR6-CT	IR4QA-CS	average
AP	aslam	**.559**/.377	**.663/.451**	.730/.561	**.818/.737**	.608/.521	**.676**/.529
	soboroff10%	.547/.396	.630/.432	**.783/.648**	**.857/.721**	**.890/.824**	**.741/.604**
	nruns30%	**.564/.398**	**.640/.447**	**.797/.659**	**.857/.714**	**.805**/.701	**.733/.584**
	sakai30%	**.561/.402**	**.633/.443**	**.800/.663**	**.851**/.704	**.818/.728**	**.733/.588**
	nuray30%	**.561/.403**	**.633/.443**	**.800/.663**	**.851**/.704	**.818/.729**	**.733/.588**
	BIAS+nuray30%	.458/.325	.509/.307	.572/.426	.737/.544	.292/.207	.514/.362
nDCG	aslam	.504/.356	.645/.425	.729/.575	.778/**.678**	.579/.504	.647/.508
	soboroff10%	.537/.383	.641/.454	.806/**.653**	**.851/.685**	**.908/.806**	.749/**.596**
	nruns30%	**.577/.436**	**.671/.490**	**.826/.695**	**.834/.682**	.892/**.785**	**.760/.618**
	sakai30%	**.572/.432**	**.664/.481**	**.819/.682**	**.826**/.673	**.903/.806**	**.757/.615**
	nuray30%	**.576/.435**	**.665/.482**	**.851/.682**	**.826**/.673	**.900/.804**	**.764/.615**
	BIAS+nuray30%	.538/.409	.535/.364	.676/.495	.774/.553	.390/.247	.583/.414

Table 4: τ and τ_{ap} rank correlations: "true" team ranking vs predicted *team* ranking. For each metric, the three highest values in each column are shown in bold (a tie is counted as one).

		ROBUST03	ROBUST04	CLIR6-JA	CLIR6-CT	IR4QA-CS	average
AP	aslam	**.517**/.271	.560/.406	**.909/.862**	**1/1**	.556/.427	.708/.593
	soboroff10%	**.500**/.374	.604/.483	**.970/.909**	**.927/.875**	**.778/.762**	**.756/.681**
	nruns30%	**.533**/.261	.538/.419	**1/1**	**.927**/.700	**.722**/.512	**.744**/.578
	TEAM+aslam	.483/.251	**.670/.535**	.758/.717	.818/**.775**	.444/.452	.635/.546
	TEAM+soboroff10%	**.517/.384**	**.670/.536**	**.909**/.847	.818/.762	**.722/.698**	**.727/.645**
	TEAM+nruns30%	**.533/.451**	**.692**/.511	.788/.747	**.855**/.625	**.667/.656**	.707/**.598**
	TEAM+spoerri	.433/**.395**	.604/**.518**	.515/.473	.600/.323	.000/−.136	.430/.315
nDCG	aslam	**.450**/.197	.516/.347	**1/1**	**1/1**	**1/1**	**.793/.709**
	soboroff10%	.433/.286	**.670**/.359	**.939/.873**	**.891/.775**	**1/1**	**.787/.659**
	nruns30%	**.500/.325**	**.670**/.366	**1/1**	.782/.575	**.889**/.708	**.768**/.595
	TEAM+aslam	**.450**/.196	.582/**.414**	.788/.733	.673/**.591**	.389/.442	.576/.475
	TEAM+soboroff10%	**.500/.373**	**.604/.404**	**.848/.788**	**.818**/.535	**.889/.896**	.732/**.599**
	TEAM+nruns30%	**.533/.512**	**.648**/.352	.818/.764	.745/.497	**.778/829**	.704/.591
	TEAM+spoerri	.367/.320	.538/**.401**	.762/.456	.455/.183	−.056/−.153	.413/.241

Observation (2) above is even more alarming: it means that the "nruns" method is as good as any. Soboroff *et al.* [16] discovered that retaining duplicates in the document pool for sampling pseudo-relevant documents is effective, which implies that documents returned by many runs are effective pseudo-relevant documents for the purpose of system ranking prediction. The "nruns" method follows this strategy, without having to retain duplicates in pools and to form 10 different pseudo-qrels. Whereas, the similarity between the "nruns30%" and "sakai30%" results imply that Sakai's second sort key (the ranks of the returned documents) does not really matter: the only useful statistic is the number of runs that returned each document. As for "nuray30%", the results are extremely similar to "sakai30%" and "nruns30%" as was predicted, and is probably not worthwhile given its computational cost.

Figure 1 visualises the run ranking errors for "aslam", "soboroff10%" and "nruns30%" with mean Q. For each data set, the runs have been sorted by the true mean Q performance: the horizontal axis represents the sorted runs; the vertical axis represents the (predicted) performance. Recall that this is more important than the correlation values. A good ranking prediction method should be smooth, and decreasing from left to right: each increase in a curve represents an error. It can be observed that the "aslam" curve for IR4QA-CS is too flat, meaning that it fails to distinguish good runs from bad ones. Whereas, it is clear that "nruns," the simplest method, behaves very similarly to Soboroff's method. The striking resemblance of the three curves highlights the fact that they all essentially rely on the same clue: the number of runs that returned each document.

Figure 2 provides scatterplots of our "nruns30%" results with mean Q. The horizontal and the vertical axis represent true and predicted mean Q, respectively. It is clear that the predicted performance values are highly correlated with the true ones. Also of interest is that, unlike what Soboroff *et al.* [16] and Aslam and Savel [3] observed, we do not see any serious underestimation of the very top performers. This may possibly be because our data sets lack truly "novel" runs: ranking based on majority votes suffices for these runs!

Of course, if there are participating runs that are very novel, the majority vote approach will probably not work. One hypothesis that may be worth testing in our future work is that while this approach may not work at early rounds of evaluation workshops where participants explore diverse techniques for a new task, it may work quite well for more mature stages of the same task (as we have witnessed in the present study), where different systems have adopted similar techniques. Such an investigation may help evaluation workshops to save cost in the long run.

Another observation from Table 3 and Figure 1 is that the NTCIR runs appear to be easier to predict than the TREC robust runs. For example, the τ values of "nruns30%" are .519-.632 in mean Q for the TREC data, and .806-.842 for the NTCIR data (not shown in Table 3). The curves for the two TREC data sets are also relatively flat, representing failure to differentiate good runs from bad ones. We shall discuss this further in Section 7.

6.2 Ranking Teams

Table 4 shows results similar to Table 3, but this time for ranking teams. One run per team was selected as described in Section 3.3 and only these runs were ranked. Since Spoerri's method relies on statistics of the selected runs only, we applied Aslam's, Soboroff's and the "nruns" methods to the same selected runs as well, to be fair to Spoerri. These runs are represented by "TEAM+aslam" and so on. In the table, "aslam," "soboroff10%" and "nruns30%" (without the TEAM prefix) represent the original methods that used the entire set of runs. Thus, the predicted scores have been ex-

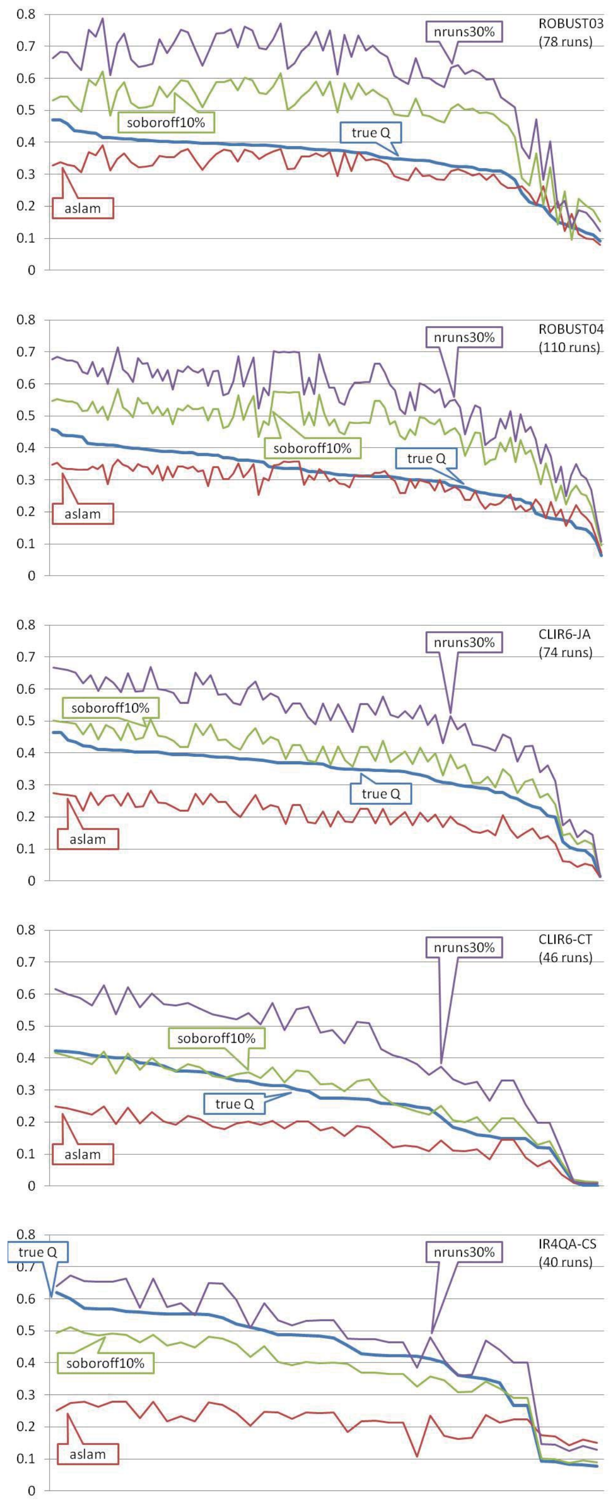

Figure 1: Accuracy of run ranking prediction. The horizontal axis represents runs sorted by true mean Q-measure values. The vertical axis represents the value of sort key for each ranking method.

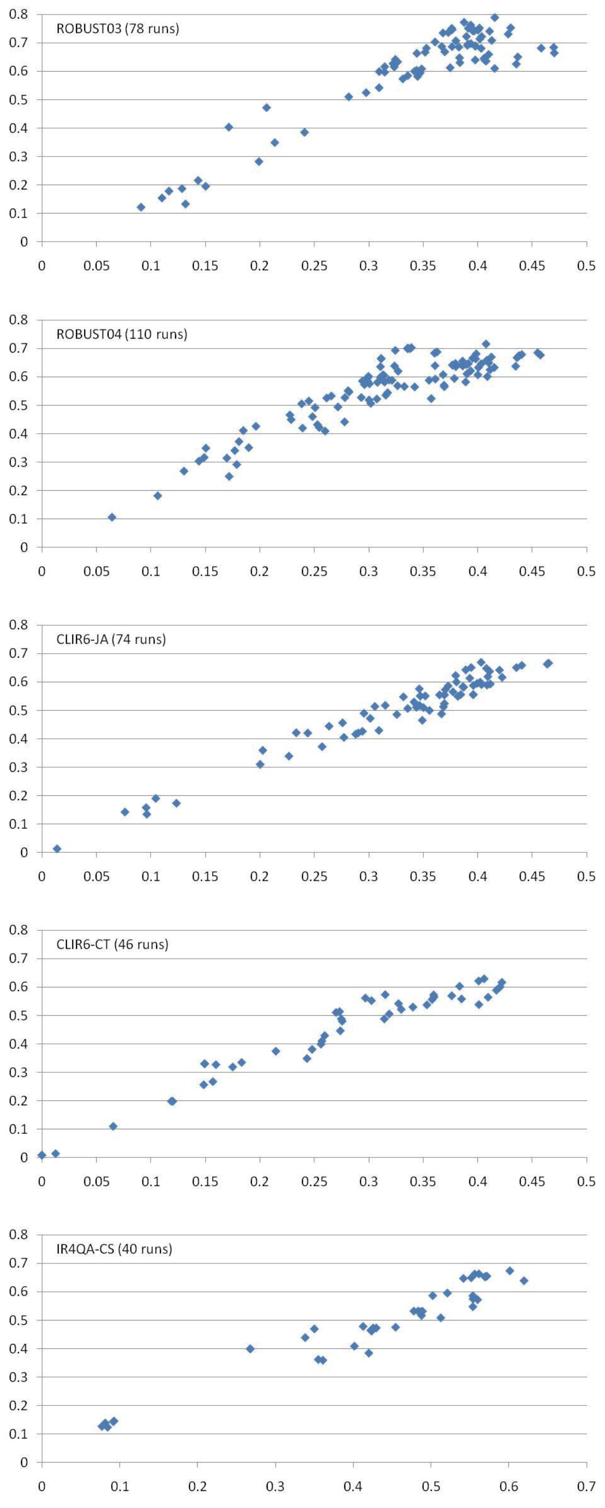

Figure 2: Scatterplot of mean Q-measure values for each run based on the "nruns30%" pseudo-qrels. The horizontal axis represents mean Q-measure computed with true qrels.

tracted from our run ranking experiments. (The results for "sakai30%" and "nuray30%" are very similar to those for "nruns30%" and are omitted.) Figure 3 visualises the team ranking results with mean Q in a way similar to Figure 1. From these results, we can observe that:

(1) Spoerri's method is not effective. While its team ranking curves do tend to resemble the other ones, it is consistently less accurate than other methods and fails completely for IR4QA-CS: see the failure to separate good teams from bad ones at the bottom of Figure 3.

(2) "aslam", "soboroff10%" and "nruns30%" are generally effective, especially for the NTCIR data. For example, "aslam" and "nruns30%" predict the mean nDCG ranking for CLIR6-JA with 100% accuracy. However, "aslam" is again a little unpredictable, in that it yields a perfectly accurate prediction for IR4QA-CS in terms of nDCG, but inaccurate prediction for the same data in terms of AP and Q.

Thus, the "nruns" method is just as effective as any other method not only for the task of ranking all runs, but also for that of ranking teams.

7. DISCUSSIONS

We have shown that, despite the number of existing publications on this topic, the number of runs that returned each document appears to be the only effective statistic for ranking systems without relevance assessments. We also showed that the NTCIR rankings tend to be easier to predict than the TREC robust track rankings. To compare TREC and NTCIR data from the viewpoint of whether documents returned by many runs do indeed tend to be relevant, we sorted the pooled documents by $|runs_t(d)|$ (i.e. the number of runs that returned each document at or above rank 30), and examined whether each document is in fact relevant. Then, for each rank in the sorted pools, the statistics were summed accross topics.

Figure 4 shows the results, with the horizontal axes representing the document ranks within the sorted pool, down to rank 150. $L0$ (in grey) represents judged nonrelevant documents. (For some of the lower ranks, the sum of judged relevant and nonrelevant docs is smaller than the topic set size, because the pool size is less than 150 for some topics.) It can be observed that, for all five test collections, documents returned by many runs are indeed more likely to be relevant than those returned by few runs. This is true even though TREC sorts pooled documents by document IDs prior to relevance assessments [19], while NTCIR sorts them either primarily or solely by $|runs_t(d)|$ [12]. Whether a sort such as that adopted by NTCIR biases human assessments [19], whether that bias (if indeed it exists) affects system ranking, and whether such a sort improves the assessment consistency and efficiency [12] are important open questions that are beyond the scope of this study. (See also footnote 2 on page 2.)

Finally, in order to further quantify the differences between the TREC robust and the NTCIR data, we tested the reliability of statistical significance test results based on the pseudo-qrels as follows: For each data set, we conducted a two-sided *bootstrap hypothesis test* with 1,000 trials [11] for every system pair from our team ranking experiments, using the "nruns" pseudo-qrels. Significantly different pairs at $\alpha = 0.05$ were recorded: let this set be C. Similarly, significantly different pairs at $\alpha = 0.05$ using the true qrels were also recorded: let this set be C_*. Then, we computed the reliability as $|C \cap C_*|/|C|$. As Table 5 shows, the reliability of the "nruns" pseudo-qrels is around 82-100% for the NTCIR data, while it is around 68-81% for the TREC robust data. Hence the TREC robust pseudo-qrels are more likely to "cry wolf."

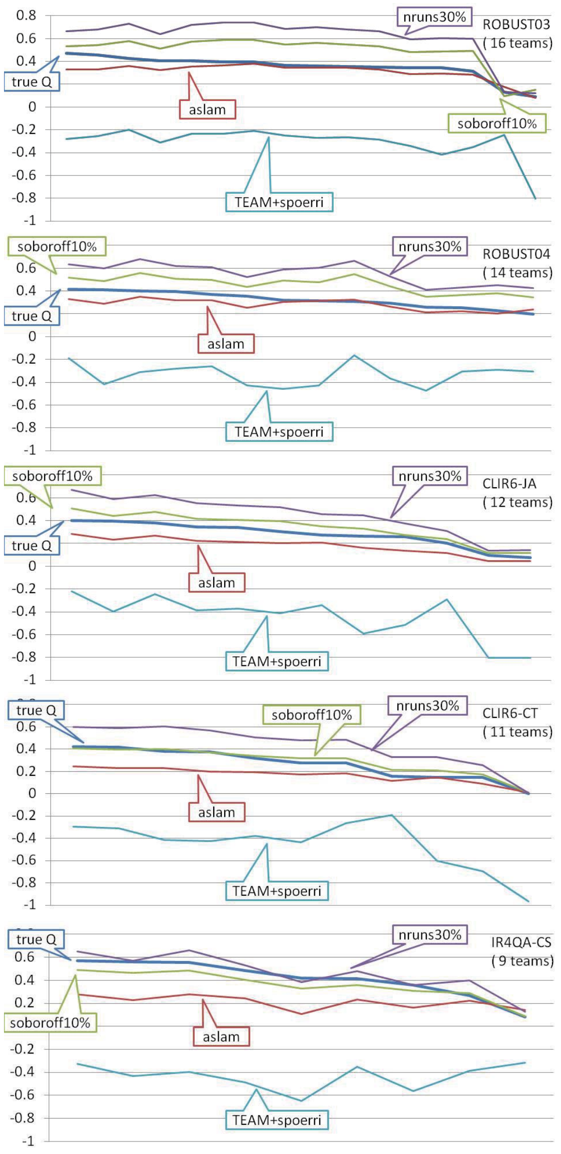

Figure 3: Accuracy of team ranking prediction. The horizontal axis represents runs (one run per team) sorted by true mean Q-measure values. The vertical axis represents the value of sort key for each ranking method.

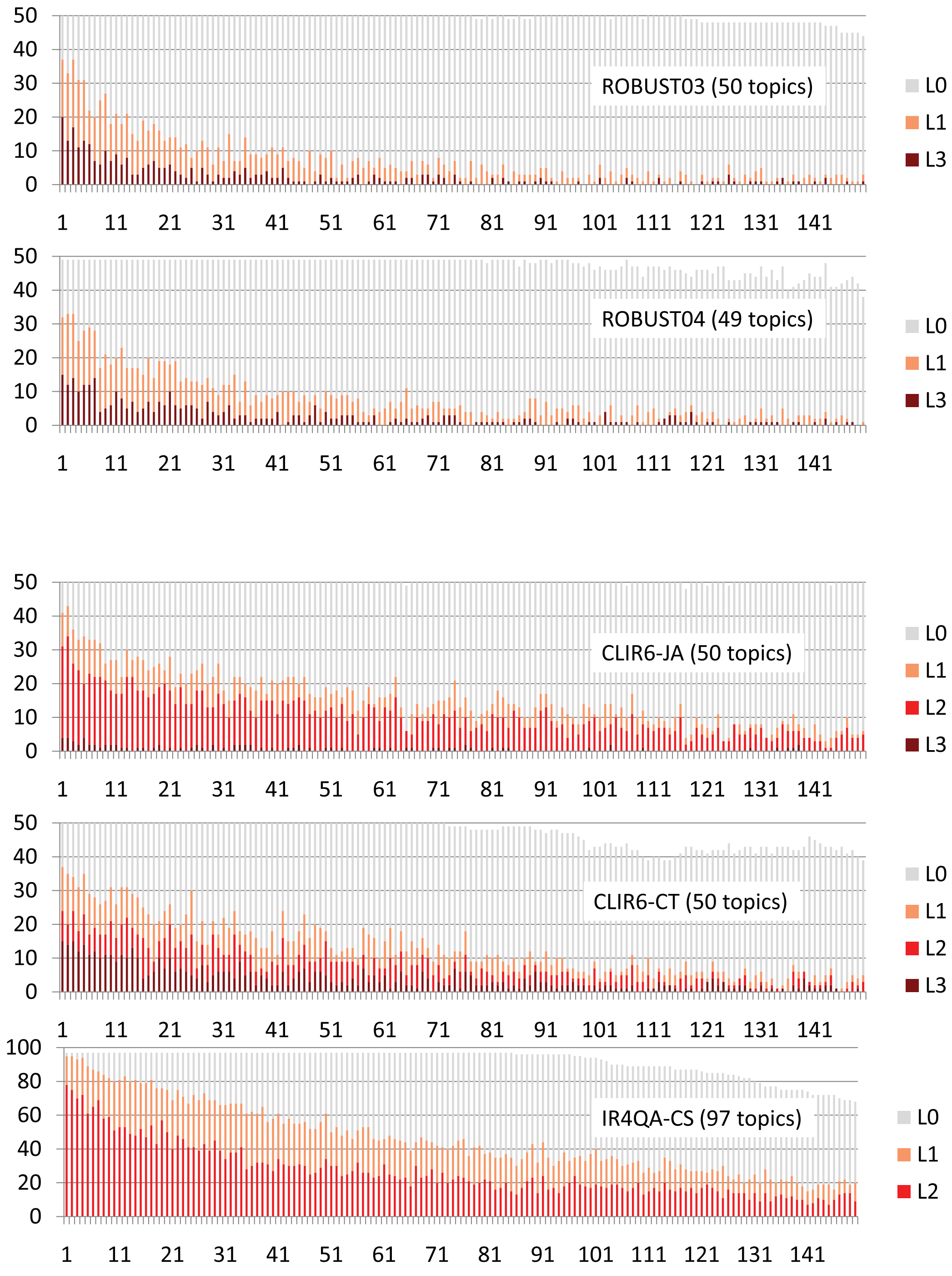

Figure 4: Number of $L3/L2/L1$-relevant and $L0$ (judged nonrelevant) documents at each rank according to $|runs_t(d)|$, summed across topics.

Table 5: Proportion of significantly different team pairs according to pseudo-qrels that are also sigfinicantly different according to true qrels.

	ROBUST03	ROBUST04	CLIR6-JA	CLIR6-CT	IR4QA-CS
AP	69/90=77%	54/69=78%	47/55=85%	44/46=96%	30/32=94%
Q	69/89=78%	56/69=81%	50/55=91%	44/46=96%	27/31=87%
nDCG	48/71=68%	54/67=81%	46/56=82%	43/43=100%	29/33=88%

8. CONCLUSIONS

Through extensive experiments on ranking systems without relevance assessments, we showed that (a) The simplest method of forming "pseudo-qrels" based on how many systems returned each pooled document performs as well as any other existing method; and that (b) the NTCIR system rankings tend to be easier to predict than the TREC robust track system rankings, and moreover, the NTCIR pseudo-qrels yield fewer false alarms than the TREC pseudo-qrels do in statistical significance testing. These differences between TREC and NTCIR may be because TREC sorts pooled documents by document IDs before relevance assessments, while NTCIR sorts them primarily by the number of systems that returned the document. However, we showed that, even for the TREC robust data, documents returned by many systems are indeed more likely to be relevant than those returned by fewer systems.

Our experimental results challenge a few previous studies [10, 17]. Lack of reproducibility and lack of "real" progress are growing concerns in the IR community [2] and elsewhere [7]. While sharing data and programs among researchers is certainly important for improving this situation, equally important are (1) describing the algorithms and experiments clearly, (2) evaluating using diverse data sets and multiple evaluation metrics. We believe that the present study has a strength over similar studies in these aspects.

Clearly, we have a long way towards semi-automatic evaluation of Web search engines, where "majority vote" is not really an option: Utilising clickthrough data (e.g. [1]), for example, is a more realistic approach for such a purpose. On the bright side, however, based on the insights from the present study, the NTCIR-8 ACLIA-2 IR4QA task has actually adopted our proposed framework of providing "system ranking forecasts" to participants right after the run submission deadline [14]. The actual usefulness of such forecasts will be investigated in future work. The relationship between the accuracy of forecasts and the maturity of evaluation workshops (see Section 6.1) should also be investigated.

9. ACKNOWLEDGMENTS

We thank Ellen Voorhees for checking for us that the TREC 2004 robust track relevance assessments were done after sorting by document IDs.

10. REFERENCES

[1] R. Agrawal, A. Halverson, K. Kenthapadi, N. Mishra, and P. Tsaparas. Generating labels from clicks. In *Proceedings of ACM WSDM 2009*, pages 172–181, 2009.

[2] T. G. Armstrong, A. Moffat, W. Webber, and J. Zobel. Improvements that don't add up: Ad-hoc retrieval results since 1998. In *Proceedings of ACM CIKM 2009*, pages 601–610, 2009.

[3] J. A. Aslam and R. Savell. On the effectiveness of evaluating retrieval systems in the absence of relevance judgments. In *Proceedings of ACM SIGIR 2003*, pages 361–362, 2003.

[4] B. Carterette. On rank correlation and the distance between rankings. In *Proceedings of ACM SIGIR 2009*, pages 436–443, 2009.

[5] B. Carterette, V. Pavlu, E. Kanoulas, J. A. Aslam, and J. Allan. Evaluation over thousands of queries. In *Proceedings of ACM SIGIR 2008*, pages 651–658, 2008.

[6] J. Guiver, S. Mizzaro, and S. Robertson. A few good topics: Experiments in topic set reduction for retrieval evaluation. *ACM Transactions on Information Systems*, 2009.

[7] J. P. Ioannidis. Why most published research findings are false. *PLoS Medicine*, 2(8), 2005.

[8] K. Kishida, K.-H. Chen, S. Lee, K. Kuriyama, N. Kando, and H.-H. Chen. Overview of CLIR task at the sixth NTCIR workshop. In *Proceedings of NTCIR-6*, pages 1–19, 2007.

[9] A. Mowshowitz and A. Kawaguchi. Assessing bias in search engines. *Information Processing and Management*, 38:141–156, 2002.

[10] R. Nuray and F. Can. Automatic ranking of information retrieval systems using data fusion. *Information Processing and Management*, 42:595–614, 2006.

[11] T. Sakai and N. Kando. On information retrieval metrics designed for evaluation with incomplete relevance assessments. *Information Retrieval*, 11:447–470, 2008.

[12] T. Sakai, N. Kando, C.-J. Lin, T. Mitamura, H. Shima, D. Ji, K.-H. Chen, and E. Nyberg. Overview of the NTCIR-7 ACLIA IR4QA task. In *Proceedings of NTCIR-7*, pages 77–114, 2008.

[13] T. Sakai, N. Kando, C.-J. Lin, R. Song, H. Shima, and T. Mitamura. Ranking the NTCIR ACLIA IR4QA systems without relevance assessments. *DBSJ Journal*, 8(2):1–6, 2009.

[14] T. Sakai, H. Shima, N. Kando, R. Song, C.-J. Lin, T. Mitamura, M. Sugimoto, and C.-W. Lee. Overview of ACLIA IR4QA. In *Proceedings of NTCIR-8*, 2010.

[15] M. Sanderson and H. Joho. Forming test collections with no system pooling. In *Proceedings of ACM SIGIR 2004*, pages 33–40, 2004.

[16] I. Soboroff, C. Nicholas, and P. Cahan. Ranking retrieval systems without relevance judgments. In *Proceedings of ACM SIGIR 2001*, pages 66–73, 2001.

[17] A. Spoerri. Using the structure of overlap between search results to rank retrieval systems without relevance judgments. *Information Proceesing and Management*, 43:1059–1070, 2007.

[18] E. M. Voorhees. Variations in relevance judgments and the measurement of retrieval effectiveness. In *Proceedings of ACM SIGIR '98*, pages 315–323, 1998.

[19] E. M. Voorhees. The philosophy of information retrieval evaluation. In *Revised Papers from the Second Workshop of CLEF (Lecture Notes in Computer Science 2406)*, pages 355–370, 2001.

[20] E. M. Voorhees. Overview of the TREC 2003 robust retrieval track. In *The Twelfth Text REtrieval Conference (TREC 2003)*, 2004.

[21] E. M. Voorhees. Overview of the TREC 2004 robust retrieval track. In *The Thirteenth Text REtrieval Conference (TREC 2004)*, 2005.

[22] S. Wu and F. Crestani. Methods for ranking information retrieval systems without relevance judgements. In *Proceedings of the 2003 ACM Symposium on Applied Computing*, pages 811–816, 2003.

[23] E. Yilmaz, J. A. Aslam, and S. Robertson. A new rank correlation coefficient for information retrieval. In *Proceedings of ACM SIGIR 2008*, pages 587–594, 2008.