Minimal Test Collections for Retrieval Evaluation

Ben Carterette
carteret@cs.umass.edu

James Allan
allan@cs.umass.edu

Ramesh Sitaraman
ramesh@cs.umass.edu

Center for Intelligent Information Retrieval
Department of Computer Science
University of Massachusetts Amherst
Amherst, MA 01003

ABSTRACT

Accurate estimation of information retrieval evaluation metrics such as average precision require large sets of relevance judgments. Building sets large enough for evaluation of real-world implementations is at best inefficient, at worst infeasible. In this work we link evaluation with test collection construction to gain an understanding of the minimal judging effort that must be done to have high confidence in the outcome of an evaluation. A new way of looking at average precision leads to a natural algorithm for selecting documents to judge and allows us to estimate the degree of confidence by defining a distribution over possible document judgments. A study with annotators shows that this method can be used by a small group of researchers to rank a set of systems in under three hours with 95% confidence.

Categories and Subject Descriptors: H.3 Information Storage and Retrieval; H.3.4 Systems and Software: Performance Evaluation

General Terms: Algorithms, Measurement, Experimentation, Theory

Keywords: information retrieval, evaluation, test collections, algorithms, theory

1. INTRODUCTION

Information retrieval system evaluation requires *test collections*: corpora of documents, sets of topics, and relevance judgments indicating which documents are relevant to which topics [15]. Ideal measures of retrieval performance should reward systems that rank relevant documents highly (*precision*) and that retrieve many relevant documents (*recall*). Stable, fine-grained evaluation metrics take both of these into account, but they require large sets of judgments to accurately measure system performance.

The TREC conferences were set up by NIST with several goals. Most relevant to this work is the goal of building test collections that could be used by information retrieval practitioners to build and evaluate their retrieval systems. The test collections need to be *reusable*; that is, they need to not only provide accurate evaluation of the set of retrieval systems submitted to TREC, but also to evaluate future retrieval systems that may be developed. To that end, NIST uses a process of *pooling* to build sets of relevance judgments: top results from many system runs on the same topics are pooled, and the entire pool is judged [11]. This gives a large set of judgments that are sufficient for future use [16].

Reusability is not always a major concern. For example, someone setting up a search engine for a collection simply wants to know which retrieval system is best for the types of queries he expects. TREC-style topics and judgments may not suit his purpose, and he may have neither the time nor money to collect the thousands of relevance judgments necessary to do a TREC-style evaluation on his own topics.

Another case is a researcher performing a user study or a preliminary investigating a new retrieval task. She wants her retrieval system to be the best possible, but she has no relevance judgments for her topics. Reusability is not a concern; she just wants to do the minimum set of judgments to tell her which system is best.

A third example is a large, highly dynamic collection such as the web. From month to month, topics and documents become obsolete; new documents enter the collection; collection statistics change. Meanwhile new algorithms must constantly be evaluated. A comparison of algorithms one month may result in a set of judgments that are not usable the next month simply because too many documents have disappeared or become less relevant as new documents and topics have appeared [9].

For any of these tasks, building a database of tens of thousands of relevance judgments is quite inefficient.

Our goal in this work is to show that it is possible and not difficult to evaluate a set of retrieval systems with high confidence with a minimal set of judgments. We present a novel perspective on average precision that leads to a natural algorithm for building a test collection. We show that average precision (AP) is normally distributed over possible sets of relevance judgments, allowing us to estimate our confidence in AP. This is used as a probabilistic stopping condition for our algorithm. In this way, evaluation and test collection construction are explicitly linked.

We then implement the algorithm and run a study with annotators. With only 15 hours of annotator time and under three hours real time we can achieve a ranking that is correct and that we have 95% confidence in. Simulation experiments test further aspects of our algorithm.

SIGIR'06, August 6–11, 2006, Seattle, Washington, USA.

2. PREVIOUS WORK

The method used by NIST to build test collections for TREC is pooling [11]. At TREC (Text REtrieval Conference) each year, sites submit retrieval runs on a corpora without relevance judgments. The top N documents from each submitted system are pooled and judged, and that set of relevance judgments is used to evaluate all systems. The pooling method has been shown to be sufficient for research purposes [16, 13].

Work building on the pooling method shows how sets of TREC judgments can be obtained more efficiently. Cormack et al. and Zobel independently investigated dynamic orderings of documents such that systems that have yielded more relevant documents are given greater priority in a pool [4, 16]. Soboroff et al. investigated random assignment of relevance to documents in a pool and found that this could actually give a decent ranking of systems [10]. More recently, Sanderson and Joho found that systems could be ranked reliably from a set of judgments obtained from a single system or from iterating relevance feedback runs [7], and Carterette and Allan proposed an algorithm based on paired comparisons of systems that was able to achieve high rank correlation with a very small set of judgments [3].

All of these works have focused on building collections from TREC systems. In this work we move away from the TREC environment to a smaller experimental environment that a research lab might find itself in.

3. INTUITION AND THEORY

Precision is the ratio of relevant documents retrieved to documents retrieved at a given rank. *Average precision* is the average of precisions at the ranks of relevant documents. For a set R of relevant documents,

$$AP = \frac{1}{|R|} \sum_{d \in R} prec@rank(d) \tag{1}$$

Average precision is a standard information retrieval evaluation metric. It has been shown to be stable [1]; that is, it reliably identifies a difference between two systems when one exists.

Let x_i be a Boolean indicator of the relevance of document i. Numbering documents in the order they were retrieved, we can rewrite AP as a sum over ranks 1 to n, the total number of documents in the corpus:

$$AP = \frac{1}{|R|} \sum_{r=1}^{n} x_r \sum_{i=1}^{r} \frac{x_i}{r} = \frac{1}{|R|} \sum_{r=1}^{n} \sum_{i=1}^{r} \frac{1}{r} x_r x_i$$

If we order documents arbitrarily, we replace the $\frac{1}{r}$ with coefficient a_{ij}:

$$AP = \frac{1}{|R|} \sum_{i=1}^{n} \sum_{j \geq i} a_{ij} x_i x_j$$

$$a_{ij} = \frac{1}{max\{rank(i), rank(j)\}}$$

To see why this is true, consider a toy example: a list of 3 documents with relevant documents x_2, x_3 at ranks 1 and 3. Average precision will be $\frac{1}{2}(\frac{1}{1}x_2^2 + \frac{1}{2}x_2x_1 + \frac{1}{3}x_2x_3 + \frac{1}{2}x_1^2 + \frac{1}{3}x_1x_3 + \frac{1}{3}x_3^2) = \frac{1}{2}(1 + \frac{2}{3})$ because $x_1 = 0, x_2 = 1, x_3 = 1$.

The difference in average precision between two systems is then

$$\Delta AP = AP_1 - AP_2 = \frac{1}{|R|} \sum_{i=1}^{n} \sum_{j \geq i} c_{ij} x_i x_j \tag{2}$$

$$c_{ij} = a_{ij} - b_{ij}$$

Suppose we believe that $\Delta AP > 0$ and we wish to find the set of documents that would prove it. Intuitively, a document that supports $\Delta AP > 0$ is relevant and has positive "weight", that is, $\sum c_{ij} x_i x_j > 0$. A document that supports the opposite hypothesis $\Delta AP < 0$ has negative weight. If we can show that the sum of the weights of relevant documents is greater than the maximum possible sum of the weights of negative documents, we can conclude that $\Delta AP > 0$.

Let S be the set of judged relevant documents and T be the set of unjudged documents. The following inequality is a sufficient stopping condition:

$$\sum_{i,j \in S} c_{ij} > \sum_{\substack{i,j \in T \text{ or} \\ i \in S, j \in T \\ \text{and } c_{ij} < 0}} |c_{ij}| \Rightarrow \Delta AP > 0 \tag{3}$$

The left-hand side (LHS) is ΔAP calculated over judged relevant documents only. The right-hand side (RHS) is an upper bound on the amount ΔAP would decrease if unjudged documents were judged relevant.

Before any documents have been judged, the LHS is 0 and the RHS is the sum of all negative coefficients. It is intuitively clear that we want to increase the LHS by finding relevant documents and decrease the RHS by finding nonrelevant documents.

3.1 An Optimal Algorithm

The stopping condition of Eq. 3 suggests an algorithm: select documents to maximize the left-hand side and minimize the right-hand side.

Suppose we have no relevance judgments. If we were to judge document i relevant, we would add c_{ii} (the difference in reciprocal ranks of i) to the LHS. If i were nonrelevant, we would subtract $c_{i1} + c_{i2} + \cdots + c_{iN} = \sum c_{ij}$ from the RHS. It seems intuitively clear that we want to pick the document that will have the greatest expected effect on either side. Give each document a "relevant weight" w_i^R (the amount it would add to the LHS if relevant) and a "nonrelevant weight" w_i^N (the amount it would subtract from the RHS if nonrelevant), and the document to judge should be the one that maximizes $\max\{p_i w_i^R, (1 - p_i) w_i^N\}$, $p_i = P(x_i = 1)$.

This algorithm is optimal for our stopping condition. We sketch part of a proof in the appendix.

3.2 AP is Normally Distributed

Let $p_i = P(x_i = 1)$. Let $q_i = 1 - p_i$. Then

$$E[AP] = \frac{1}{\sum p_i} \sum_{i}^{n} \left(a_{ii} p_i + \sum_{j > i} a_{ij} p_i p_j \right) + \epsilon \tag{4}$$

$$Var[AP] = \frac{1}{(\sum p_i)^2} \left(\sum_{i}^{n} a_{ii}^2 p_i q_i + \sum_{j > i} a_{ij}^2 p_i p_j (1 - p_i p_j) + \sum_{i \neq j} 2 a_{ii} a_{ij} p_i p_j q_i + \sum_{k > j \neq i} 2 a_{ij} a_{ik} p_i p_j p_k q_i \right) + \epsilon$$

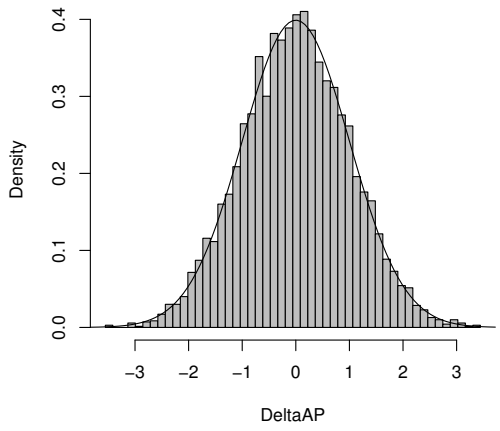

Figure 1: Average Precision is normally distributed. We simulated two ranked lists of 100 documents. Setting $p_i = .5$, we randomly generated 5000 sets of relevance judgments and calculated ΔAP for each set. The histogram conforms to a standard normal distribution. The Anderson-Darling goodness of fit test [12] concludes that we cannot reject the hypothesis that the sample came from a normal distribution.

The error term $\epsilon = O(c^{-n})$ for some $c > 1$, so we may safely ignore it.[1]

As $n \to \infty$,

$$\frac{AP - E[AP]}{\sqrt{Var[AP]}} \rightsquigarrow N(0,1)$$

This means that with any incomplete test collection, AP is distributed normally over all possible assignments of relevance to all unjudged documents. We have shown this by simulation (Fig. 1), and have sketched a formal proof using martingale limit theory. Eq. 4 is normally distributed independent of how a_{ij} is defined, so ΔAP is distributed normally as well.

Given a set of relevance judgments, we use the normal cumulative density function (cdf) to find $P(\Delta AP \leq 0)$. If $P(\Delta AP \leq 0) < .05$, at least 95% of the possible assignments of relevance would conclude that $\Delta AP > 0$ and we can conclude that $\Delta AP > 0$ with 95% confidence. Our previous stopping condition is replaced by the probabilistic stopping condition $P(\Delta AP \leq 0) < 1 - \alpha$ `OR` $P(\Delta AP \leq 0) > \alpha$.

Note that we must make some assumption about p_i. We make the "neutral" assumption that if i is unjudged, $p_i = 0.5$.Using prior information about rates of relevance or ranks at which documents were retrieved could give better results, but we do not explore that here.

3.3 Application to MAP

Mean average precision is the average of APs computed for a set of topics T. Because topics are independent,

$$\mathcal{E}MAP = \frac{1}{|T|} \sum_{t \in T} E[AP_t]$$

$$Var[MAP] = \frac{1}{|T|^2} \sum_{t \in T} Var[AP_t]$$

Then if $AP \rightsquigarrow N(0,1)$, $MAP \rightsquigarrow N(0,1)$ as well.

The algorithm for determining a difference ΔMAP follows directly from the algorithm for average precision. We treat each (topic, document) pair as a unique "document" and proceed exactly the same way.

4. EXPERIMENTAL EVALUATION

Having developed the algorithm from first principles, we wish to show that it can be used in a research setting on a new corpus. Previous work has shown simulated results of efficient judging algorithms on TREC ad hoc collections [3, 4, 7, 10, 16]; rather than repeat those experiments, we will evaluate the algorithm as it might be used in a research or industrial environment.

The outline of our experiment is as follows: we ran eight retrieval systems on a set of baseline topics for which we had full sets of judgments. We asked six annotators to develop new topics; these were run on the same eight systems. The same six annotators then judged documents selected by the algorithm.

4.1 Baseline Topics

The baseline topics were used to estimate the system performance. It is generally assumed in Information Retrieval that 50 topics are a sufficient sample for reliable comparison of retrieval systems. This has been shown recently by Sanderson and Zobel [8] and earlier by Zobel [16]. We expect, therefore, that performance of systems on our new topics will track performance on the baseline topics.

The baseline topics were the 2005 Robust/HARD track topics [14] and ad hoc topics 301 through 450. We used only topic titles for queries.

4.2 Corpora

We indexed two different corpora: one was the Aquaint corpus consisting of about 1 million articles from the New York Times News Service, Associated Press Worldstream News Service, and Xinhua News Service from 1996 through 2000. The second was TREC disks 4 and 5, with about 500,000 documents from the Financial Times, Federal Register, LA Times, and Foreign Broadcast Information Service, covering 1989 through 1996. We ran the Robust queries on the Aquaint corpus and the ad hoc queries on the TREC corpus, so even though some topics were duplicated, the retrieved results are different.

4.3 Retrieval Systems

We used six freely-available retrieval systems: Indri, Lemur, Lucene, mg, SMART, and Zettair[2]. With these six systems

[1] $E[AP]$ and $Var[AP]$ are actually sums over exponentially many terms. These approximations are intuitive, and the error is negligible.

[2] Respectively available at:
http://www.lemurproject.org/indri
http://www.lemurproject.org
http://lucene.apache.org
http://www.cs.mu.oz.au/mg
ftp://ftp.cs.cornell.edu/pub/smart
http://www.seg.rmit.edu.au/zettair

	Topics						
System	All 200	Robust 05	301-350	351-400	401-450	60 New	60 New-norm
1	.109	.103	.129	.093	.112	.162±.007	.118
2	.123	.110	.148	.108	.124	.158±.007	.100
3	.168	.144	.174	.165	.189	.174±.008	.187
4	.169	.162	.179	.150	.184	.175±.008	.192
5	.172	.141	.166	.174	.208	.179±.008	.218
6	.213	.194	.233	.177	.249	.185±.008	.246
7	.215	.203	.231	.182	.244	.187±.008	.260
8	.296	.321	.300	.262	.301	.194±.008	.300

Table 1: True MAPs of eight systems over 200 topics; broken out into four sets of 50 topics; and expected MAP, with 95% confidence intervals, over 60 new topics. The final column is expected MAP rescaled to the range $[.1, .3]$ for easy comparison to performance on other topic sets. Horizontal lines indicate "bin" divisions determined by statistical significance.

we obtained eight retrieval runs on all 200 baseline queries. The runs use different retrieval models (Okapi weighting, TFIDF weighting, language modeling), different stemmers (Krovetz, Porter), and different stop lists. One run used pseudo-relevance feedback.

Since we did not tune the systems, we want to avoid identifiable claims about relative performance. We number the systems 1 through 8, with 1 having lowest mean average precision on all 200 baseline topics and 8 having highest.

Results of the eight retrieval runs are shown in Table 1. The table shows that the ranking can vary some by topic sets, though the bins the systems fall into remain the same. Significant differences between pairs of systems are always preserved.

4.4 Query Formulation

Six volunteer annotators were given an interface to query and browse the Aquaint corpus and asked to come up with 10 topics each. They were asked to provide a title query and a description of what should and should not be considered relevant. They were asked to create topics that were not too easy (too many relevant documents found in the browsing interface), but not too hard (no relevant documents).

We wanted to minimize drift in annotators' definitions of relevance. We kept the time between defining topics and judging documents as short as possible so that annotators would not forget what they were thinking. All judging was done in a 3-hour block with a break for lunch so that definitions of relevance would be unlikely to change drastically. Our annotators were experienced in the field of information retrieval; we hoped their greater understanding of the definition of relevance would also help keep drift down. Finally, by asking annotators to explicitly state what should be considered relevant and displaying that to them while they judged documents, we hoped to reduce drift further.

Some of the topics (along with number of judgments made and number of relevant documents found) are shown in Table 2. Apart from two topics about "intelligent design", all topics were unique.

4.5 Algorithm Implementation

The algorithm was implemented in `R` [5]. Using vector arithmetic, weights could be calculated in linear time; the complexity of the algorithm is $O(S^2n^2)$. We used only the top 100 documents retrieved by each system, partially for computational reasons, but also because this is the usual cutoff for collecting judgments for TREC. If a document was not ranked in the top 100 by a system, its reciprocal rank for that system was defined to be 0.

No.	title query	judged	rel
6	environmental conservation policies	39	25
13	journalistic plagiarism	31	10
24	sea piracy	55	29
31	intelligent design	43	16
36	indian nuclear tests and visas	49	7
44	africa aids orphans	92	75
54	environmental impact of US army	27	2
59	aquifer water levels	12	0

Table 2: Selected topics developed by annotators.

When comparing pairs of systems, we used the document with max weight in the pair. To extend that to ranking a set of systems, we use the document with the max weight in all pairs of systems.

The implementation was very fast. There was no perceivable lag between an annotator submitting a judgment and receiving the next document.

Strictly speaking, the algorithm requires that judgments be made one at a time in the determined order. This would require that annotators spend a lot of time idle, waiting for a document from one of their topics to be served. Instead, we separated the topics into six sets of ten and ran the algorithm independently on each set. This may give slightly suboptimal performance, but it is a better use of annotator time.

4.6 Experimental Process

The 60 topics developed were run on the same eight retrieval systems against the Aquaint corpus. A server was set up to feed documents to annotators for judging. They used a web interface to judge documents. Each annotator judged documents from their own 10 topics. They spent about 2.5 hours judging (15 total hours of annotator time), with a break for lunch after the first hour.

5. RESULTS

In 2.5 hours we obtained 2200 relevance judgments, about 4.5 per system per topic on average, about 2.5 per minute per annotator, and about 14.7 per minute. The TREC pool-

ing approach with a depth of 100 would have yielded 18,537 documents for judgment. NIST annotators spent a total of 280.5 hours judging the 37,798 documents in the pool for Robust/HARD systems, a rate of 2.2 judgments per minute.

Of the documents judged, 38.5% were relevant. Judgments by annotators on the first 271 documents were compared to judgments by one of the authors of this paper; annotator agreement was 88%.

The fastest annotator judged twice as many documents as the slowest. Some topics were more difficult to interpret than others, and some topics retrieved documents that were longer on average; that affected the number of documents an annotator could judge. In section 6 we simulate a single annotator judging all topics.

We rank systems by expected value of MAP

$$\mathcal{E}MAP = \sum_t E[AP_t] = \sum_t \frac{1}{\sum p_i} \sum_i \left(c_{ii} p_i + \sum_{j>i} c_{ij} p_i p_j \right)$$

where $p_i = 1$ if document i has been judged relevant, 0 if nonrelevant, and .5 otherwise.

$\mathcal{E}$MAP ranges from 0 to 1. A ranking by $\mathcal{E}$MAP is directly comparable to a ranking by MAP, though $\mathcal{E}$MAP has compressed range compared to MAP. $\mathcal{E}$MAP is shown in Table 1 with the MAPs on the baseline topics for comparison. The ranking is within the range of normal variation we would expect from performance on different topic sets. In fact, apart from the inversion of the first two systems, the ranking is identical to the canonical ranking. Table 1 also shows $\mathcal{E}$MAP rescaled to the range $[0.1, 0.3]$ for comparison to the other topics.

Using the normal cdf, we calculate the confidence that $\Delta MAP > 0$ for each pair of systems. We estimate the confidence in the ranking as the average of the confidences in each pair. With no judgments, confidence is .5; with a full set of judgments, confidence would be 1. This can be interpreted as the expected confidence in ΔMAP if any two systems are selected from the set at random. By that measure, our algorithm has 96% confidence that this is the correct ranking.

Figure 2 shows how confidence increases as more judgments are made. A correct ranking is found very fast: after only 50 judgments, the systems are binned correctly; after 250 judgments, the systems are ranked correctly. Confidence after 250 judgments is only 71%, but after 1000 judgments, the systems are ranked correctly with 90% confidence. It took about one hour real time (6 hours of annotator time) to reach that point.

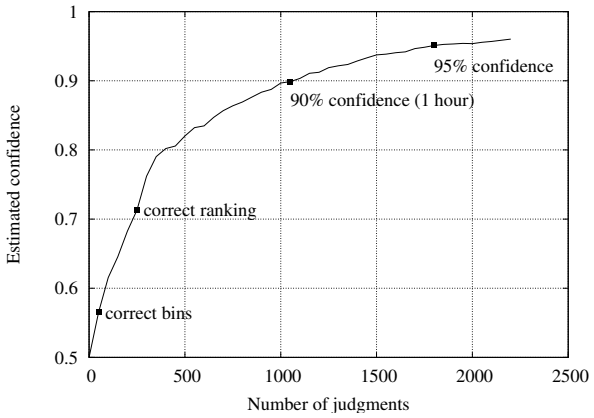

Figure 2: Confidence increases as more judgments are made.

	2	3	4	5	6	7	8
1	.007 .480	.040 **0**	.059 **0**	.037 **0**	.090 **0**	.100 **0**	.217 **0**
2	—	.034 **0**	.052 **0**	.031 **0**	.084 **0**	.093 **0**	.211 **0**
3		—	.018 .061	.003 .144	.050 **.002**	.060 **0**	.177 **0**
4			—	-.021 .793	.032 .073	.041 **.029**	.159 **0**
5				—	.053 **.018**	.063 **.003**	.180 **0**
6					—	.010 .185	.127 **0**
7						—	.117 **0**

Table 3: Each cell shows difference in true MAP and $P(\Delta MAP \leq 0)$ for each pair of systems after 2200 judgments for the Robust 2005 topics. Pairs in which we have 95% confidence that additional judgments will not change the outcome are bolded.

6. DISCUSSION

Here we present some results of simulations of our algorithm to attempt to evaluate its performance in general settings. The simulation is done by running the client/server setup described previously, but using judgments from the *qrels* produced by NIST instead of judging manually. Documents not judged by NIST were assumed nonrelevant. Table 3 shows true ΔMAP and $P(\Delta\mathcal{E}\text{MAP} \leq 0)$ for a simulation using the Robust 2005 topics.

Some questions we explore are:

- To what degree are the results dependent on the algorithm rather than the evaluation metric?
- How many judgments are required to differentiate a single pair of ranked lists with 95% confidence?
- How does confidence vary as more judgments are made?
- Are test collections produced by our algorithm reusable?

6.1 Comparing $\mathcal{E}$MAP and MAP

The main advantage of $\mathcal{E}$MAP over standard MAP is that it takes advantage of information provided by nonrelevance. MAP is calculated using only relevant documents, and thus it only helps to find relevant documents (except insofar as a nonrelevant judgment indicates that the document has been judged and does not need to be judged again). More value is extracted from each judgment.

Another advantage is that $\mathcal{E}$MAP is normally distributed. This provides all the power that the normal distribution provides. The caveat is that we must make an assumption about the probability of a document's relevance.

We ran the following simulation: after several documents had been judged, we calculated both $\mathcal{E}$MAP and MAP on all systems, ranked them, and compared the ranking to the "true" ranking by MAP computed using all relevance judg-

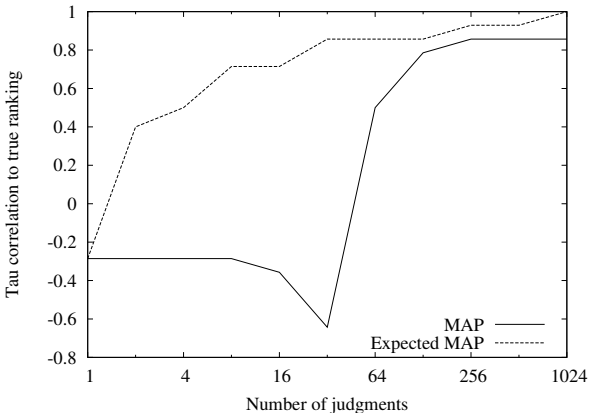

Figure 3: As the number of judgments increases, Kendall's tau correlation to the true ranking increases. Correlation between Expected MAP and true MAP increases faster than correlation between MAP and true MAP.

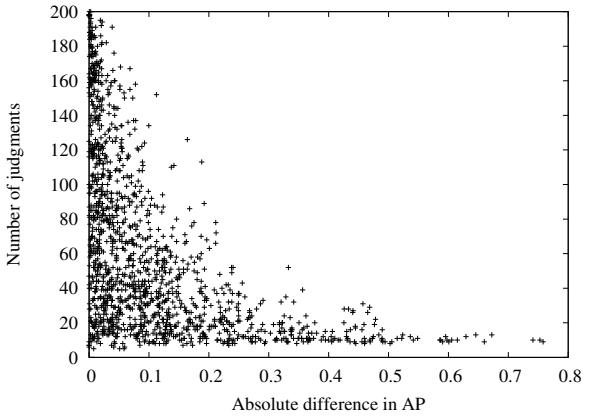

Figure 4: Absolute difference in true AP and the number of judgments it takes to achieve 95% confidence that a difference exists. Each point represents a pair of ranked lists. In 98% of the pairs, the sign of the difference was predicted correctly.

ments. We did this after 1, 2, 4, 8, ..., 1024 judgments.

To compare rankings we use a rank correlation measure called *Kendall's tau* [6]. Kendall's tau is calculated using pairwise inversions between elements in two lists. It ranges from -1 to 1, with -1 indicating that the two lists are reversed, 0 that half the pairs of elements are inverted, and 1 that the two lists are identical. Since we have only 8 systems, the range of values is limited.

The result is shown in Figure 3. We see that the correlation between $\mathcal{E}$MAP and true MAP increases very fast; after only 32 judgments it is about .85. It takes regular MAP 256 judgments to get to the same point.

The *bpref* metric also uses nonrelevance information [2]. This result tracks with bpref.

6.2 How Many Judgments?

The number of judgments that must be made in comparing two systems or a set of systems depends on how similar the systems are. "Similarity" could be measured by true difference in average precision, or by a ranking distance metric.

Figure 4 shows absolute difference in true AP for Robust 2005 topics vs. number of judgments to 95% confidence for pairs of ranked lists for individual topics. As the figure shows, if difference in AP is greater than .25, it will generally take fewer than 25 judgments to distinguish performance. But as difference in AP gets closer to 0, the number of judgments rises fast. The axes can be flipped: if many documents have been judged but 95% confidence is not in sight, the lists must be very similar. The correlation between difference in AP and number of judgments to 95% confidence is -0.509.

A simple distance measure is the sum over all documents of absolute difference in reciprocal rank, i.e. $d = \sum |\frac{1}{r_i} - \frac{1}{r'_i}|$. The correlation between distance and number of judgments is 0.218.

Difference in AP and our distance metric have a correlation of 0.219, indicating that an increase in distance leads to an increase in difference in AP.

6.3 Confidence over Time

Figure 5 shows a long simulation of a single annotator judging documents for the Robust/HARD queries. Confidence reaches .95 very rapidly, about as fast as it did in Figure 2, and after that continues to approach 1, the point at which the ranking would not change with more judgments. It takes 16,000 judgments to reach 1, less than half the number in the Robust 2005 *qrels*. The curve from Figure 2 is included for comparison.

Figure 6 shows a comparison to a pooling method that we will call "incremental pooling". In this method all documents in a pool of depth k will be judged. Since the order in which they are judged affects the estimated confidence, we must impose an ordering on the pool. We judge them in rank order. This could be seen as a weaker version of our algorithm, which will tend to judge high-ranked documents unless they are found at a similar position in every list.

The pool of depth 10 for our Robust/HARD runs contains 2228 documents. There were 569 documents in the pooled set that were not in our algorithmic set (and vice versa). This means that our algorithm tends to pick documents from the top of the ranked lists, which is expected since those have the greatest effect on MAP. It also shows that high confidence can be achieved more rapidly by drawing documents from outside the pool.

6.4 Reusability of Test Collection

We showed above that $\mathcal{E}$MAP can produce a good ranking with a very small set of judgments. This suggests that a test collection created by this algorithm may be able to evaluate new systems that were not used in the creation of the test collection.

To test this, we removed one system from our set of eight and simulated our algorithm to build test collections of 500, 1000, 1500, and 2000 judgments from the remaining seven. Then we replaced the eighth system and ranked all eight by $\mathcal{E}$MAP, setting p_i to be the ratio of relevant documents in the test collection.

Table 4 shows rank confidence averaged over eight trials, removing a different system for each trial. Confidence ac-

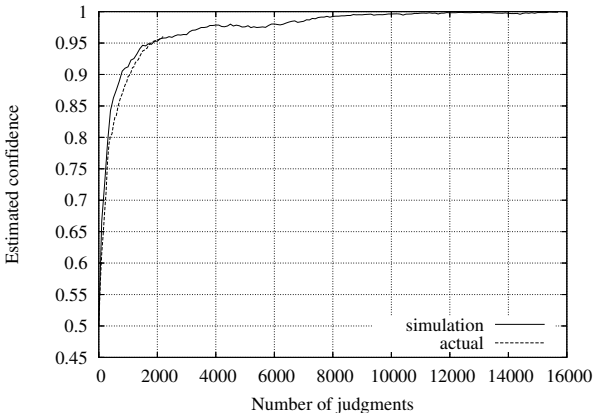

Figure 5: A long simulation showing confidence approaching 1. The original experiment is shown for comparison.

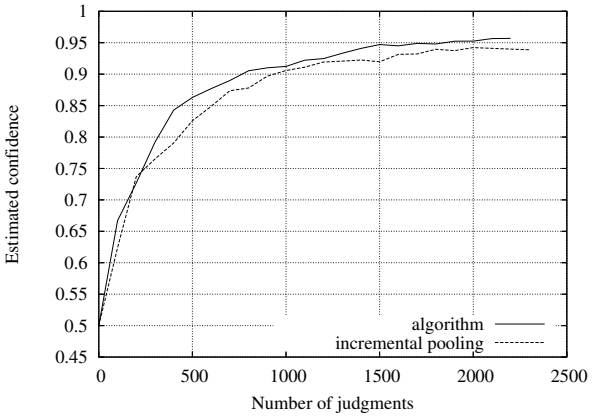

Figure 6: Comparison between our algorithm and incremental pooling. We have more confidence in the rankings by the algorithm's documents than by pooling's.

tually *increases* when adding the 8th system back in, likely because of the better estimate of p_i. Furthermore, the 8th system is always placed in the correct spot in the ranking or swapped with the next (statistically indistinguishable) system. This suggests that the test collections can be reused reliably in at least some cases. In general it depends on how many documents in the new system have been judged and the estimates of p_i.

7. CONCLUSION

A new perspective on average precision leads to an algorithm for selecting documents that should be judged to evaluate retrieval systems in minimal time. Using actual annotators, we implemented the algorithm and showed that it can be used to rank retrieval systems with a high degree of confidence and a minimal number of judgments. After only six hours of annotation time, we had a achieved a ranking with 90% confidence. This is applicable to a variety of retrieval environments when little relevance information is available.

no. judgments	Rank confidence 7 systems	8 systems
500	.863	.881
1000	.921	.930
1500	.945	.954
2000	.955	.969

Table 4: Reusability of test collections. Judgments are collected from seven systems, then all eight are ranked using those judgments. Rank confidence increases when the eighth is replaced.

A clear direction for future work is extending the analysis to other evaluation metrics for different tasks. Some initial exploration in this direction suggests that it would be very easy to analyze precision, for instance. Another direction is estimating probabilities of relevance. Uniformly using the same value is not optimal; better estimates based on ranks provide better results.

Acknowledgments

This work was supported in part by the Center for Intelligent Information Retrieval, in part by the Defense Advanced Research Projects Agency (DARPA) under contract number HR001-06-C-0023, and in part by an NSF award under grant number CNS-0519894. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsor.

8. REFERENCES

[1] C. Buckley and E. M. Voorhees. Evaluating Evaluation Measure Stability. In *Proceedings of SIGIR*, pages 33–40, 2000.

[2] C. Buckley and E. M. Voorhees. Retrieval evaluation with incomplete information. In *Proceedings of SIGIR*, pages 25–32, 2004.

[3] B. Carterette and J. Allan. Incremental Test Collections. In *Proceedings of CIKM*, pages 680–687, 2005.

[4] G. V. Cormack, C. R. Palmer, and C. L. Clarke. Efficient Construction of Large Test Collections. In *Proceedings of SIGIR*, pages 282–289, 1998.

[5] R. Gentleman and R. Ihaka. The R Language. In *Proceedings of the 28th Symposium on the Interface*, 1997.

[6] M. Kendall. *Rank Correlation Methods*. Griffin, London, UK, fourth edition, 1970.

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

[8] M. Sanderson and J. Zobel. Information retrieval system evaluation: Effort, sensitivity, and reliability. In *Proceedings of SIGIR*, pages 186–193, 2005.

[9] A. Singhal. Challenges in Running a Commercial Search Engine. In *Proceedings of SIGIR*, page 432, 2005. Keynote Talk.

[10] I. Soboroff, C. Nicholas, and P. Cahan. Ranking Retrieval Systems without Relevance Judgments. In *Proceedings of SIGIR*, pages 66–73, 2001.

[11] K. Sparck Jones and C. J. van Rijsbergen. Information Retrieval Test Collections. *Journal of Documentation*, 32(1):59–75, 1976.

[12] M. A. Stephens. EDF Statistics for Goodness of Fit and Some Comparisons. *Journal of the American Statistical Association*, 69:730–737, 1974.

[13] E. Voorhees. Variations in Relevance Judgments and the Measurement of Retrieval Effectiveness. In *Proceedings of SIGIR*, pages 315–323, 1998.

[14] E. Voorhees. Overview of the TREC 2005 Robust Retrieval Track. In *TREC 2005 Notebook*, 2005.

[15] E. M. Voorhees. The philosophy of information retrieval evaluation. In *CLEF '01: Revised Papers from the Second Workshop of CLEF*, pages 355–370, London, UK, 2002. Springer-Verlag.

[16] J. Zobel. How Reliable are the Results of Large-Scale Information Retrieval Experiments? In *Proceedings of SIGIR*, pages 307–314, 1998.

APPENDIX

In Section 2 we stated a sufficient stopping condition (Eq. 3) that would allow us to conclude $\Delta AP > 0$. Here we present an algorithm for maximizing the expected value of part of the stopping condition and prove its optimality. This is only part of the algorithm we actually use in the paper; a complete algorithm and proof are forthcoming.

$$\sum_{i,j \in S} c_{ij} x_i x_j > \sum_{i \notin S \text{ or } j \notin S; c_{ij} < 0} |c_{ij}| \Rightarrow \Delta AP > 0 \quad (5)$$

This is a restatement of the stopping condition Eq. 3: S is a set of documents that have been judged. Intuitively, we want to show that our algorithm will select documents that maximize the expected value of the left-hand side and minimize the expected value of the right-hand side. We claim Algorithm 1 maximizes the expected value of the LHS when $p_i = p_j = p$ for all i, j.

Algorithm 1 Select k documents to be judged.

```
1: while |S| < k do
2:   for all i ∉ S do
3:     w_i ← c_ii p + Σ_{j∈S} c_ij p^2
4:   end for
5:   j ← argmax_i w_i
6:   S ← S + {j}
7: end while
```

THEOREM 1. *If $p_i = p$ for all i, the set S maximizes $E[\sum_{i,j \in S} c_{ij} x_i x_j]$.*

To prove this, we will need to show that the weight of S is greater than the weight of any other set of the same size.

We begin with two lemmas:

LEMMA 1. *For two documents i and j,*

$$c_{ii} \geq c_{ij} \geq c_{jj}$$
$$\text{or} \quad c_{jj} \geq c_{ij} \geq c_{ii}$$

PROOF. The proof is by a simple analysis of cases. There are four possible relative orderings of i and j in two systems, and in every case one of the above inequalities is true. □

LEMMA 2. *At iteration i of the algorithm, $c_{ij}p^2 \geq c_{jj}p^2$ for $j \geq i$.*

PROOF. By induction on i. The base case, $c_{1j} \geq c_{jj}$, follows because $c_{11} \geq c_{jj}$ by lines 3 & 5 of Alg. 1, and $c_{11} \geq c_{1j} \geq c_{jj}$ by Lemma 1.

Assume the induction hypothesis: $c_{i-1,j} \geq c_{jj}$ for arbitrary $i-1$. The proof continues by contradiction. Suppose $c_{jj} > c_{ij}$. If this is true, then anywhere we place an arbitrary document k we have that $c_{jk} \geq c_{ik}$ (this can be shown by enumeration of nine cases, some of which are impossible because they violate the induction hypothesis). This means that $c_{jj}p + \sum_k c_{jk}p^2 > c_{ii}p + \sum_k c_{ik}p^2$. But this is a contradiction: we selected document i because $c_{ii}p + \sum_k c_{ik}p \geq c_{jj}p + \sum_k c_{jk}p$ for all $j > i$ (lines 3 & 5 in Alg. 1). Therefore we must conclude that $c_{ij} \geq c_{jj}$. □

Now we are ready to prove the theorem. The proof is fairly technical, but the intuition is that, given an arbitrary set U and the set S produced by Algorithm 1, we show that the expected weight of S is greater than the expected weight of a set S' that contains a particular document in U instead of the kth document in S, and that the expected weight of S' is greater than the expected weight of U.

PROOF. By induction on k.

The base case is trivial: $c_{11}p \geq c_{ii}p$ by construction.

Let S_k be a set of k documents. Let U_k be an arbitrary set of k documents such that $U_k \neq S_k$. We shall number documents in the order they would be added to S, so U_k will contain documents with indices greater than k, but S_k will not. For conciseness, the expected weight of a set S $E[\sum_{i,j \in S} c_{ij} x_i x_j]$ shall be denoted $\sum S$. Let us make the induction hypothesis that $\sum S_{k-1} \geq \sum U_{k-1}$. We wish to show that $\sum S_k \geq \sum U_k$.

Note that $S_{k-1} \subset S_k$. Therefore $\sum S_k = \sum S_{k-1} + c_{kk}p + \sum_{j \in S_{k-1}} c_{jk}p^2$. Let ℓ be the document in U_k such that $\ell \geq k$ and ℓ has greater difference in reciprocal ranks than any other document $u \geq k$, i.e. $c_{\ell\ell} \geq c_{uu}$ for all $u \geq k$. Lemma 1 ensures ℓ exists by imposing an ordering on the differences in reciprocal ranks c_{ii} for all i. Lemma 2 states that $c_{j\ell}p^2 \geq c_{\ell\ell}p^2$ for $j < \ell$. Then it follows that $c_{j\ell}p^2 \geq c_{\ell\ell}p^2 \geq c_{u\ell}p^2 \geq c_{uu}p^2 \geq c_{uv}p^2 \geq c_{vv}p^2$....

Let S'_k be the set obtained by replacing document k with document ℓ defined above, i.e. $S'_k = S_k - k + \ell$. $\sum S_k \geq \sum S'_k$ by construction. If we remove ℓ from both S'_k and U_k, we have $S'_k - \ell = \sum S_{k-1} \geq U_k - \ell$ by the induction hypothesis. It remains to be shown that $\sum_{j \in S_{k-1}} c_{j\ell}p^2 \geq \sum_{u \in U_k - \ell} c_{u\ell}p^2$. Let j be an arbitrary document in S_{k-1}. If there is a $u \in U_k - \ell$ such that $j = u$, then $c_{j\ell}p^2 = c_{u\ell}p^2$. Otherwise, $c_{j\ell}p^2 \geq c_{\ell\ell}p^2 \geq c_{u\ell}p^2$.

Then

$$\sum S_k \geq \sum S'_k = \sum S_{k-1} + c_{\ell\ell}p + \sum_{j \in S_{k-1}} c_{j\ell}p^2$$
$$\geq \sum (U_k - \ell) + c_{\ell\ell}p + \sum_{u \in U_k - \ell} c_{u\ell}p^2 = \sum U_k$$

and the proof is complete. □

A similar algorithm is used to pick nonrelevant documents to decrease the right hand side of Eq. 5. Instead of $w_i = c_{ii}p + \sum_{j \in S} c_{ij}p^2$, it sets $w_i = c_{ii}(1-p) + \sum c_{ij}(1-p)^2$. Similar reasoning proves that the set selected has minimal expected weight.