Million Query Track 2008 Overview

James Allan*, Javed A. Aslam†, Ben Carterette‡, Virgil Pavlu†, Evangelos Kanoulas†

* Department of Computer Science
University of Massachusetts Amherst, Amherst, Massachusetts

† College of Computer and Information Science
Northeastern University, Boston, Massachusetts

‡ Department of Computer and Information Sciences
University of Delaware, Newark, Delaware

The Million Query (1MQ) track ran for the second time in TREC 2008. The track is designed to serve two purposes: first, it is an exploration of ad-hoc retrieval over a large set of queries and a large collection of documents; second, it investigates questions of system evaluation, in particular whether it is better to evaluate using many shallow judgments or fewer thorough judgments.

As with the 2007 track [ACA$^+$07], participants ran 10,000 queries against a collection of 25 million documents. The 2008 track differed in the following ways:

1. Queries were assigned to one of four categories.

2. Each query was assigned a target of 8, 16, 32, 64, or 128 judgments.

3. Assessors could judge documents "not relevant but reasonable".

Section 1 describes how the corpus and queries were selected, the query classes, details of the submission formats, and a brief description of each submitted run. Section 2 provides an overview of the judging process, including a sketch of how it alternated between two methods for selecting the small set of documents to be judged. Sections 3.1 and 3.2 provide an overview of those two selection methods, developed at UMass and NEU, respectively.

In Section 4 we present statistics collected during the judging process, including the total number of queries judged, how many judgments were served by each approach, and so on, along with the overall results of the track. We present additional results and analysis in Section 5.

1 Phase I: Running Queries

The first phase of the track required that participating sites submit their retrieval runs.

1.1 Corpus

The 1MQ track used the so-called "terabyte" or "GOV2" collection of documents. This corpus is a collection of Web data crawled from Web sites in the .gov domain in early 2004. The collection is believed to include a large proportion of the .gov pages that were crawlable at that time, including HTML and text, plus the extracted text of PDF, Word, and PostScript files. Any document longer than 256Kb was truncated to that size at the time the collection was built. Binary files are not included as part of the collection, though were captured separately for use in judging.

The GOV2 collection includes 25 million documents in 426 gigabytes. The collection was made available by the University of Glasgow, distributed on a hard disk that was shipped to participants for an amount intended to cover the cost of preparing and shipping the data.

1.2 Queries

Topics for this task were drawn from a large collection of queries that were collected by a large Internet search engine. Each of the chosen queries is likely to have at least one relevant document in the GOV2 collection because logs showed a clickthrough on one page captured by GOV2. Obviously there is no guarantee that the clicked page is relevant, but it increases the chance of the query being appropriate for the collection.

These topics are short, title-length (in TREC parlance) queries. In the judging phase, they were developed into full-blown TREC topics.

Ten thousand (10,000) queries were selected for the official run. Queries were categorized based on length and number of clicks on documents in the .gov domain. Half of the 10,000 had no more than six words (designated "short") and half more than six words (designated "long"); half had more than three clicks in the .gov domain ("govheavy") and half three or fewer ("govslant"). The exact distribution is shown in Table 1. (The numbers in the table does not add to 10,000 because 264 of the queries were designated for use with the Relevance Feedback track and excluded from Million Query judging.)

	short	long
govslant	2,434	2,434
govheavy	2,434	2,434

Table 1: Distribution of queries by category: length at most six words ("short") versus more than six ("long"), and at most three clicks on GOV2 documents ("govslant") versus more than three ("govheavy").

No quality control was imposed on the 10,000 selected queries. The hope was that most of them would be good quality queries, but it was recognized that some were likely to be partially or entirely non-English, to contain spelling errors, or even to be incomprehensible to anyone other than the person who originally created them.

The queries were distributed in a text file where each line has the format "N:query word or words". Here, N is the query number, is followed by a colon, and immediately followed by the query itself. For example, the line "32:barack obama internships" means that query number 32 is the 3-word query "barack obama internships". All queries were provided in lowercase and with no punctuation. Query categories were not provided.

1.3 Submissions

Sites were permitted to submit up to five runs. Every submitted run was included in the judging pool and all were treated equally. In addition, there were five "baseline" runs reflecting standard retrieval models.

A run consisted of up to the top 1,000 documents for each of the 10,000 queries. The submission format was a standard TREC format of exactly six columns per line with at least one space between the columns. For example:

```
100 Q0 ZF08-175-870 1 9876 mysys1
100 Q0 ZF08-306-044 2 9875 mysys2
```

where:

1. The first column is the topic number.
2. The second column is unused but must always be the string "Q0" (letter Q, number zero).
3. The third column is the official document number of the retrieved document, found in the <DOCNO> field of the document.
4. The fourth column is the rank of that document for that query.
5. The fifth column is the score this system generated to rank this document.
6. The six column was a "run tag," a unique identifier for each group and run.

If a site would normally have returned no documents for a query, it instead returned the single document "GX000-00-0000000" at rank one. Doing so maintained consistent evaluation results (averages over the same number of queries) and did not break any evaluation tools being used.

1.4 Submitted runs

The following is a brief summary of some of the submitted runs, from the summary information provided by the submitting sites.

ARSC/University of Alaska Fairbanks submitted five runs.

lsi150dyn The documents from each unique host name were grouped and indexed as separate collections. Each host collection was represented as a "big document" in a vector space of approximately 400,000 terms. Each topic was projected into the term vector space. The host collections were ranked relative to each topic-vector using a 150-rank LSI-reduced host-by-term matrix. The topic was run against the top 50 ranked hosts with Lucene and the ranked results from each host were merged using a standard result set merge algorithm.

lsi150stat The documents from each unique host name were grouped and indexed as separate collections. Each host collection was represented as a "big document" in a vector space of approximately 400,000 terms. The 10,000 topics were represented as a "big topic" in the term space. The host collections were ranked relative to this static "big topic" using an 150-rank LSI-reduced host-by-term matrix and the 10,000 topics were run against the top 50 most relevant hosts and the results merged into a single ranked list.

vsmdyn The documents from each unique host name were grouped and indexed as separate collections. Each host collection was represented as a “big document” in a vector space of approximately 400,000 terms. Each topic was projected into the term vector space. The host collections were ranked relative to each topic-vector using vector space model cosines with the host-by-term matrix. The topic was run against the top 50 ranked hosts with Lucene and the ranked results from each host were merged using a standard result set merge algorithm.

vsmstat The documents from each unique host name were grouped and indexed as separate collections. Each host collection was represented as a “big document” in a vector space of approximately 400,000 terms. The 10,000 topics were represented as a “big topic” in the term space. The host collections were ranked relative to this static “big topic” using standard vector space model (VSM) cosines with the host-by-term matrix; the 10,000 topics were run against the top 50 most relevant hosts and the results merged into a single ranked list.

vsmstat07 This run is to test the sensitivity of the static “big topic” vector space ranking (restriction) of host collections searched. In short, the hosts were chosen by ranking them with respect to a “big topic” from the 10000 TREC 2007 MQ topics using the vector space model as in vsmstat. Then the TREC 2008 MQ topics were run against this collection of hosts (that was tuned for the 2007 topics).

I3S Group of ICT submitted two runs.

dxrun We use Wikipedia as a resources to identify entities in a query, then add term dependency features for each query. The term dependency features are actually ordered phrases. Indri search engine is used for index and retrieval.

txrun We use Wikipedia as a resources to identify entities in a query, then expand each query with ten terms(if there are any) based on Wikipedia. The term selection procedure makes use of semantic similarity measure proposed by resnik(1996). Terms are ranked in descending order according to the similarity between a term and the who query. Indri search engine is used for indexing and retrieval.

IBM Haifa submitted two runs.

LucDeflt Lucene run with default settings (*) Default similarity - that is default doc length normalization = 1/sqrt(num tokens) and default tf = sqrt(term freq) (*) Single field for all searchable text (*) No special query parts (no phrases and no proximity scoring) (*) Default OR operator.

LucLpTfS Lucene run with proximity and phrase scoring, doc length normalization by Lucene’s sweet-spot similarity, and tf normalization by average term frequency.

Max Planck Institute D5 submitted one run.

mpiimq0801 standard BM25, no stemming, standard stopword removal.

Northeastern University submitted four runs.

hedge0 We used several standard Lemur built in systems (tfidf_bm25, tfidf_log, kl_abs, kl_dir, inquery, cos, okapi) and combined their output (metasearch) using the hedge algorithm.

neuMSRF Uses MSLIVE on the WWW to generate a query "context" like pseudo-relev feedback (50 terms selected). Then tfidf formula is used only for feedback terms (original query terms are not used) against GOV2 index. Uses Lucene toolkit.

neumsfilt In the first phase, each query is run through MS Live Search. Then the top 10 documents and snippets are use to generate pseudo-relevant feedback, so more words are added/re-weighted to the query. Finally, we run okapi for query terms. For each document that okapi considers, a bonus score is given if the document contains more than some threshold of words identified as interesting from the MS live feedback set. Uses Lucene toolkit.

neustbl BM25 on Lucene toolkit, with modifications on query preprocessing. Two sets of documents are considered: the ones with all query words (if none, then consider the documents with maximum number of query words) , and the ones with at least one query word. The union of these two sets are ranked using Okapi BM25. Uses Lucene toolkit.

SabIR submitted two runs.

sabmq08a1 Standard smart ltu.Lnu run.

sabmq08b1 SMART blind feedback, top 25 docs, add 20 terms.

University of Massachusetts Amherst submitted four runs.

ind25QLnST08 Standard query-likelihood, no stopping.

indriQLST08 Standard query-likelihood with a standard stopword list.

indriLowMu08 Query-likelihood with Dirichlet smoothing parameter mu set to 1. No stopping.

indri25DM08 Dependence model approach fielded during the Terabyte track two years ago and the 1MQ track last year.

In addition, there were four "baseline" runs of standard retrieval algorithms, all using the Lemur implementations: a vector-space cosine similarity run (000cos), a language modeling run (000klabs), an Okapi formula run (000okapi), a BM25 run with tf-idf weighting (000tfidfBM25), and a log tf-idf run (000tfidfLOG).

2 Phase I: Relevance judgments and judging

After all runs were submitted, a subset of the topics were judged. The goal was to provide a small number of judgments for a large number of topics. For TREC 2008, 782 queries were judged, a large increase over the more typical 50 queries judged by other tracks in the past, though a significant decrease from the 1700 judged for TREC 2007.

2.1 Judging overview

Judging was done by assessors at NIST. Participating sites were not asked to provide judgments this year. The process looked roughly like this from the perspective of someone judging:

1. The assessment system presented 10 queries randomly selected from one of the four categories (short-govslant, short-govheavy, long-govslant, or long-govheavy). The category from which queries were drawn rotated in a round-robin fashion by assessor to ensure that each category would be benefit from roughly the same judging effort.

2. The assessor either selected one of those ten queries to judge, or refreshed the display to see a new list of 10 from the same category. Once a query was chosen, all others that were seen were returned to the pool.

3. The assessor provided the description and narrative parts of the query, creating a full TREC topic. This information was used by the assessor to keep focus on what is relevant.

4. The system presented a GOV2 document (Web page) and asked whether it was relevant to the query. Judgments were on a four-way scale: highly relevant, relevant, not relevant but reasonable, or not relevant. Consistent with past practice, the distinction between the first two was up to the assessor.

   The "not relevant but reasonable" judgment is new to the 2008 track. It was a concession to the fact that a short query is vague, and after seeing retrieved documents, assessors may realize that there are other possible topic definitions that are as reasonable as the one they submitted. Assessors used this judgment to indicate a document that was not relevant to the topic they submitted, but would be relevant to a reasonable alternative.

5. The assessor was required to continue judging until a pre-defined stopping point was reached. Each query was judged to a target of 8, 16, 32, 64, or 128 judgments. The target was selected by a scheduling algorithm that ensured that the total amount of assessor effort at each target would be roughly equal. For every query with a target of 128, there were two with a target of 64, four with a target of 32, eight with a target of 16, and 16 with a target of 8. Because of overlap between the two methods (see below), the total number of judgments obtained could be less than the intended target.

The system for carrying out those judgments was built at UMass on top of the Drupal content management platform[1]. The same system was used for the Relevance Feedback track and as the starting point for relevance judgments in the Enterprise track.

2.2 Selection of documents for judging

Two approaches to selecting documents were used:

Minimal Test Collection (MTC) method. In this method, documents are selected by how much they inform us about the difference in mean average precision given all the judgments that were made up to that point [CAS06]. Because average precision is quadratic in relevance judgments, the amount each relevant document contributes is a function of the total number

[1] http://drupal.org

of judgments made and the ranks they appear at. Nonrelevant documents also contribute to our knowledge: if a document is nonrelevant, it tells us that certain terms in the quadratic expansion cannot contribute anything to average precision. We quantify how much a document will contribute if it turns out to be relevant or nonrelevant, then select the one that we expect to contribute the most. This method is further described below in Section 3.1.

Statistical evaluation (statMAP) method. This method draws and judges a specific random sample of documents from the given ranked lists and produces unbiased, low-variance estimates of average precision, R-precision, and precision at standard cutoffs from these judged documents [AP07]. Additional (non-random) judged documents may also be included in the estimation process, further improving the quality of the estimates. This method is further described below in Section 3.2.

Each query was judged by alternating between the two methods until each had selected half of the target number of judgments. For example, if the target was eight, the first document might be selected by MTC, the second by statAP, the third by MTC, and so on. (The method to select the first document was determined by a coin flip.) The two methods had no knowledge of judgments to each other's documents during this phase. If a document was selected by one method after it had previously been selected by the other and judged by the assessor, it was not judged a second time. Instead, the judging system supplied the same judgment that had previously been made. Thus it was possible that fewer documents could be judged than were originally targeted.

3 Methods

For a full description of the two methods, we refer the reader to the 2007 Million Query Track overview [ACA+07] and the original work cited below. The descriptions below focus on estimates of R-precision and precision at rank thresholds that are new to the 2008 track, in addition to estimates of mean average precision that we have used in previous work.

3.1 Minimal Test Collections

The Minimal Test Collections (MTC) method works by identifying documents that will be most informative for understanding performance differences between systems by some evaluation measure (in this case average precision). Details on the workings of MTC can be found elsewhere [CPK+08, CAS06, ACA+07]. Here we focus on MTC estimates of evaluation measures.

First, we consider each document i to have a distribution of relevance $p(X_i)$. If the document has been given judgment $j = 0$ or 1 (nonrelevant or relevant), then $p(X_i = j) = 1$; otherwise the probability that the document is relevant is $p(X_i = 1) = p_i$. Then we consider a measure to be a random variable expressible as a function of document relevances X_i. For example, precision can be expressed as a random variable that is a sum of document relevance random variables for those documents at ranks 1 through k.

If the measure is a function of document random variables, then the measure has a distribution over possible assignments of relevance to unjudged documents. Note that this applies to measures averaged over queries as well as to measures for a single query.

Abstracting even higher, this produces a distribution over possible rankings of systems. There is some probability that system 1 is better than system 2, which in turn is better than system 3;

some probability that system 1 is better than system 3, which in turn is better than system 2, and so on. It can be shown that the maximum a posteriori ranking of systems is that in which systems are ranked by the expected values of the evaluation measure of interest over the possible relevance assignments.

Calculating the expectation of an evaluation measure is fairly simple. Given the probability that document i is relevant $p_i = p(X_i = 1)$, we define:

$$\mathbf{E}prec@k = \frac{1}{k}\sum_{i=1}^{k} p_i$$

$$\mathbf{E}R\text{-}prec \approx \frac{1}{\mathbf{E}R}\sum_{i=1}^{\mathbf{E}R} p_i$$

$$\mathbf{E}AP \approx \frac{1}{\mathbf{E}R}\sum_{i=1}^{n} p_i/i + \sum_{j>i} p_i p_j/j$$

$$\text{and } \mathbf{E}R = \sum_{i=1}^{n} p_i.$$

Note that there is an assumption that document relevances are independent, or conditionally independent given features used to estimate the distribution $p(X_i)$.

Though MTC is designed for ranking systems (i.e. making decisions about relative differences in performance between systems), In this work we largely present expectations of evaluation measures for individual systems.

3.2 statAP

In statistical terms, average precision can be thought of as the mean of a population: the elements of the population are the relevant documents in the document collection and the population value of each element is the precision at this document for the list being evaluated. This principle is the base for several recently proposed evaluation techniques [YA06, APY06, AP, ACA$^+$07]. StatAP is a sample-and-estimate technique defined by the following two choices.

Stratified Sampling, as developed by Stevens [BH83, Ste], is very straightforward for our application. Briefly, it consists of bucketing the documents ordered by a chosen prior distribution and then sampling in two stages: first sample buckets with replacement according to cumulative weight; then sample documents inside each bucket without replacement according to selection at the previous stage.

Generalized ratio estimator. Given a sample S of judged documents along with inclusion probabilities, in order to estimate average precision, *statAP* adapts the generalized ratio estimator for unequal probability designs [Tho92].(very popular on polls, election strategies, market research etc). For our problem, the population values are precisions at relevant ranks; so for a given query and a particular system determined by ranking r(.) we have (x_d denotes the relevance judgment of document d) :

$$statAP \quad = \quad \frac{1}{\widehat{R}}\sum_{d\in S}\frac{x_d \cdot \widehat{prec@r}(d)}{\pi_d}$$

where

category	8	16	32	64	128	total
short-govslant	95 (7.87)	55 (15.58)	29 (29.93)	13 (58.85)	4 (117.50)	196 (18.92)
short-govheavy	118 (7.85)	40 (15.18)	26 (30.27)	10 (58.60)	3 (117.67)	197 (16.54)
long-govslant	98 (7.72)	52 (15.60)	26 (30.38)	13 (58.31)	8 (116.88)	197 (20.56)
long-govheavy	92 (7.79)	57 (15.32)	21 (29.95)	14 (59.29)	10 (114.40)	194 (21.61)
total	403 (7.81)	204 (15.43)	102 (30.14)	50 (58.78)	25 (116.08)	784 (19.40)

Table 2: Number of queries judged per category and target, with number of judgments per query in parentheses.

$$\widehat{R} = \sum_{d \in S} \frac{x_d}{\pi_d} \; ; \qquad \widehat{prec@k} = \frac{1}{k} \sum_{d \in S, r(d) \le k} \frac{x_d}{\pi_d}$$

are estimates the total number of relevant documents and precision at rank k, respectively, both using the Horwitz-Thompson unbiased estimator [Tho92].

Confidence intervals. We can compute the inclusion probability for each document (π_d) and also for pairs of documents (π_{df}); therefore we can calculate an estimate of variance, $\widehat{var}(statAP)$, from the sample, using the ratio estimator variance formula found in [Tho92], pp. 78 (see [AP, ACA+07] for details). Assuming the set of queries Q is chosen randomly and independently, and taking into account the weighting scheme we are using to compute the final MAP (see Results), we compute an estimator for the MAP variance

$$\widehat{var}(statMAP) = \frac{1}{(\sum_q w_q)^2} \sum_{q \in Q} w_q^2 \cdot \widehat{var}(statAP_q)$$

where w_q is distribution weight proportional with the number of judgments made on query q. Assuming normally distributed $statMAP$ values, a 95% confidence interval is given by $\pm 2std$ or $\pm 2\sqrt{\widehat{var}(statMAP)}$.

4 Experiment and Results

4.1 Queries and judgments

In total we obtained 15,211 judgments for 784 queries. Of these, 747 (5%) were "not relevant but reasonable", 2,001 (13%) were "relevant", and 931 (6%) were "highly relevant"; the rest were not relevant.

As described above, queries were partitioned into four categories (overlapping) and assigned a target number of judgments to ensure roughly equal judging effort for each category and each target. Table 2 shows the distribution of judged documents broken out by category and by judgment target. Multiplying the number of queries by the number of judgments per query shows that there are roughly the same number of judgments for each category and each target. It is true that the number of judgments at each target decreases as the target increases; this is because the probability of collisions increases with the target, from 2.3% ($= 100 \cdot 7.81/8$) at target 8 to 9.3% ($= 100 \cdot 116.08/128$) at target 128. Nevertheless, the number of collisions is quite low, suggesting that the two methods are identifying very different sets of documents as important.

Table 3 shows the proportion of documents marked relevant broken out by query category and maximum number of judgments. Note that the "govheavy" categories had a significantly

category	8	16	32	64	128	total
short-govslant	0.1872	0.1214	0.2028	0.1399	0.0298	0.1459
long-govslant	0.2021	0.1702	0.1734	0.1201	0.1369	0.1597
short-govheavy	0.2462	0.3081	0.3037	0.2338	0.3739	0.2832
long-govheavy	0.2887	0.2039	0.2226	0.1361	0.1600	0.1958
total	0.2313	0.1928	0.2251	0.1524	0.1575	0.1928

Table 3: Percent of documents judged relevant.

greater proportion of documents judged relevant than the "govslant" categories. The difference between "short" and "long" is not as clear, but we can see from the table that the increase from "short-govslant" to "short-govheavy" is quite a bit larger than the increase from "long-govslant" to "long-govheavy".

The length of judged documents varied by category. Measuring length by number of characters (a loose measure that also includes HTML tags, Javascript, metadata, and more that would not be visible to the user), documents judged for short queries had an average length of 38,730 characters, while those judged for long queries had an average length of 43,900 characters. There is a smaller difference for govslant and govheavy: an average length of 40,456 characters for the former, and 42,175 for the latter.

4.2 Assessor Time

We measured the elapsed time assessors spent performing various interactions with the judging system. Assessor time was mainly spent in three areas:

1. selecting a query
2. backfitting the query to a topic definition
3. judging documents

When selecting a query, assessors were allowed to refresh the page to see a new list of 10. They could do this as many times as they wanted. How many times did they refresh? How long did they look at each list of 10? How long did they look at the final list of 10 before selecting one? And then how long did they spend developing it into a topic?

These numbers are shown in Table 4 along with the time spent on topic definition. Note that all numbers are median seconds—the mean is inappropriate because the distribution is heavily skewed by breaks. On average, the assessors looked at 2.413 pages of queries before selecting one—meaning they picked roughly one out of every 24.13 queries they saw. Three of the assessors looked at about two pages of queries every time (on average). One assessor looked at an average of more than 7 pages of queries! This suggests that assessors were willing to spend some time looking for queries that they felt comfortable with.

Assessors looked at each page of 10 queries for 22 seconds (median). There was a large amount of variance in this number, with one assessor looking at each page for only 13 seconds, and three looking at each page for about 35 seconds. Before choosing a query, they looked at the last page of 10 queries for 29 seconds (median), slightly longer than the average look. After selecting a query, they took 76 seconds (median) to develop it into a topic. It is possible that the looking time at

					judgment times					
category	refresh	view	lastview	topic	8	16	32	64	128	average
short	2.34	18.0	25.5	67.6	15.0	11.5	13.5	12.0	8.5	12.5
long	2.54	24.5	31.0	86.5	17.0	14.0	16.5	10.0	10.5	13.0
govslant	2.22	22.5	29.0	76.0	13.0	12.5	13.0	9.5	10.5	12.0
govheavy	2.65	20.0	27.5	78.0	19.0	13.0	17.0	12.5	8.5	13.5
average	2.41	22.0	29.0	76.0	15.0	13.0	15.0	11.0	9.0	13.0

Table 4: Average number of query lists viewed and median seconds spent viewing each list, viewing the final list, and defining the topic for the selected query.

the last page of queries was slightly longer because they began mentally formulating a topic before clicking on a query. It is very clear from the table that long queries resulted in more time spent selecting a query and defining a topic. This may explain why long queries seemed to produce more stable rankings: the topic definitions may have been more carefully constructed and relevance judgments were more consistent. There is a hint that less time was spent on govheavy queries than govslant.The number of refreshes does not vary a great deal by category.

Time to make judgments is also shown in Table 4. Here too time varied by query type, and somewhat surprisingly, by the total number of judgments. The fact that each judgment was made faster when 128 were requested suggests that assessors may have some "ramp-up" time on a topic, after which they can make judgments much faster, though it may also suggest that they become less careful in their judging over time.

Did certain types of judgments take longer than others? We calculated the median number of seconds to make a "nonrelevant", "not relevant but reasonable", "relevant" and "highly relevant" judgment. Respectively, it took 10 seconds, 41 seconds, 23 seconds, and 27 seconds. Recall that assessors could reformulate their topic in between judgments. As it turns out, the proportion of "reasonable" judgments that were immediately preceded by a topic reformulation was greater than that for any other type of judgment: 6.2% for reasonable compared to 5.7% for relevant, 5.1% for highly relevant, and 1.9% for nonrelevant. In addition, assessors spent more time on the reformulations that were made before a reasonble judgment than they did for any other type. A working hypothesis is that certain documents compelled assessors to "tighten" their definition of relevance, and that these occurred more often closer to the boundary between relevant and not relevant.

4.3 Results

We evaluated each system by MTC and statAP measures over all queries; in addition, each system was evaluated over each category individually and each judgment target individually:

1. all queries, all judgments

2. queries with [8|16|32|64|128] judgments

3. long/short or govslant/govheavy queries

weighted MAP. Both evaluation methods estimate average precision, for each run, for each query. Our overall baseline measurement is mean average precision (MAP); however, since the queries have various number of documents judged, we compute a *weighted MAP* over all queries for each run r (separately for each evaluation method):

$$wMAP(r) = \sum_q AP_q(r) w_q$$

where w_q is proportional with the number of documents judged for query q, and $\sum_q w_q = 1$. The weighted MAP formula essentially counts each judgment made uniformly, (or equivalently, each of the 5 level-judgment classes uniformly), because it can be written as the average of straight MAP for each judgment level class c

$$wMAP(r) = \frac{1}{5}\sum_{j=1}^{5} MAP_j = \frac{1}{5}\sum_{j=1}^{5}\frac{1}{Q_j}\sum_{q\in j} AP_q$$

where MAP_j is averaged over all queries with target j ($= 2^{j+2}$ target judgments) and Q_j is the number of queries at target j. The weighted MAP is a compromise between the TREC ad-hoc tracks setup, where few queries are judged heavily, and the Million Query 2007 track, where many queries were judged lightly.

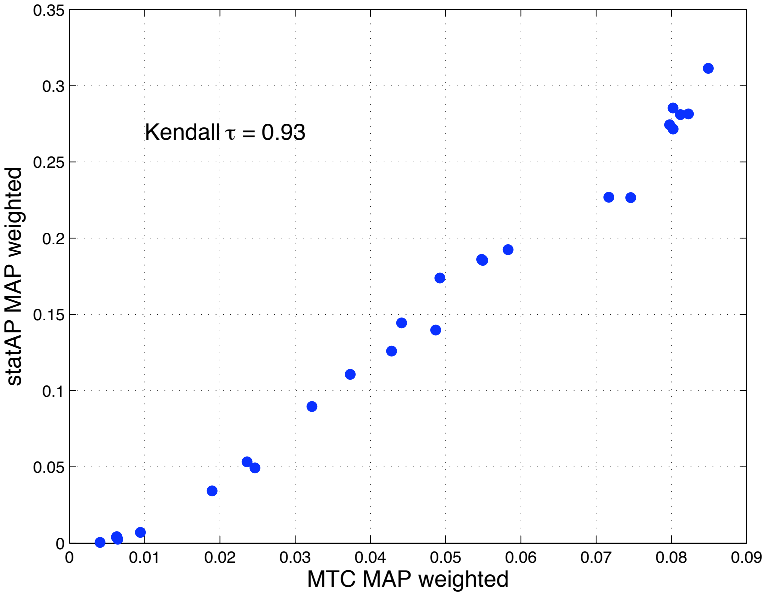

Figure 1: MTC and statAP weighted-MAP correlation

In the absence of any traditional evaluation baseline (where a lot more documents are judged for each query), the best indication that our rankings are close to the "true" ranking of systems is the correlation of the two evaluation methodologies. Their mechanisms for estimation are fundamentally different, so any correlation is much more likely to be due to correct estimation than any other reason. Figure 1 illustrates the weighted MAP scatterplot, with a Kendall τ=.93. It turns out that for this experiment in particular, there is virtually no difference in rankings obtained (for

run	%unjudged	statAP wMAP	MTC wMAP	statAP CI	MTC conf
hedge0	0.9971	0.0004	0.0041	±0.0003	1.000
vsmstat07	0.9536	0.0039	0.0062	±0.0078	0.869
vsmstat	0.9549	0.0043	0.0063	±0.0078	0.911
lsi150stat	0.9571	0.0025	0.0064	±0.0029	1.000
sabmq08a1	0.9758	0.0069	0.0094	±0.0128	1.000
lsi150dyn	0.9684	0.0342	0.0190	±0.0179	1.000
000cos	0.9596	0.0533	0.0236	±0.0096	0.529
vsmdyn	0.9642	0.0493	0.0247	±0.0184	1.000
000tdfLOG	0.9606	0.0896	0.0322	±0.0163	1.000
000tdfBM25	0.9579	0.1106	0.0373	±0.0148	1.000
000klabs	0.9446	0.1259	0.0428	±0.0159	1.000
000okapi	0.9412	0.1443	0.0441	±0.0160	1.000
sabmq08b1	0.9630	0.1398	0.0487	±0.0200	0.994
LucDet	0.9399	0.1739	0.0492	±0.0150	1.000
indriLowMu08	0.9371	0.1861	0.0548	±0.0133	0.997
mpiimq0801	0.9498	0.1855	0.0549	±0.0204	1.000
neuMSRF	0.9665	0.1925	0.0583	±0.0198	1.000
neumslt	0.9517	0.2266	0.0746	±0.0239	1.000
neustbl	0.9384	0.2268	0.0717	±0.0207	1.000
dxrun	0.9334	0.2744	0.0798	±0.0191	0.547
txrun	0.9345	0.2854	0.0802	±0.0200	0.560
indriQLST08	0.9285	0.2716	0.0803	±0.0191	0.951
ind25QLnST08	0.9312	0.2810	0.0812	±0.0204	0.583
LucLpTfS	0.9412	0.2815	0.0823	±0.0224	1.000
indri25DM08	0.9330	0.3114	0.0849	±0.0210	NA

Table 5: MTC and statAP estimate of MAP. Numbers are weighted averages of AP query estimates. The confidence in statAP MAP estimate is given as a 95% confidence interval (column 5); the confidence of MTC (column 6) is the probability that the system's performance is lower than the one of the next system.

each evaluation method) with straight MAP vs weighted MAP: for MTC the rankings are correlated with $\tau = 0.97$ and for statAP the correlation is $\tau = .99$ (Figure 2). We can get a deeper sense of the agreement by breaking queries out into subsets by category and judgment count. The number of queries in each category is roughly the same (see Table 2). MTC and statAP continue to agree well, with τ correlations of 0.93, 0.82, 0.89, and 0.92 for short-govslant, short-govheavy, long-govslant, and long-govheavy, respectively.

The two methods continue to agree highly even when evaluated over only the documents they selected ($\tau = 0.87$), and even when evaluated over only the documents selected by *the other method* ($\tau = 0.91$)—despite very little overlap between the methods. Showing that the two methods agree very highly on the ranking of systems even when evaluating over disjoint query/judgment sets provides very strong evidence that these methods are finding something close to the "true" ranking of systems, and possibly the actual true ranking modulo assessor disagreement.

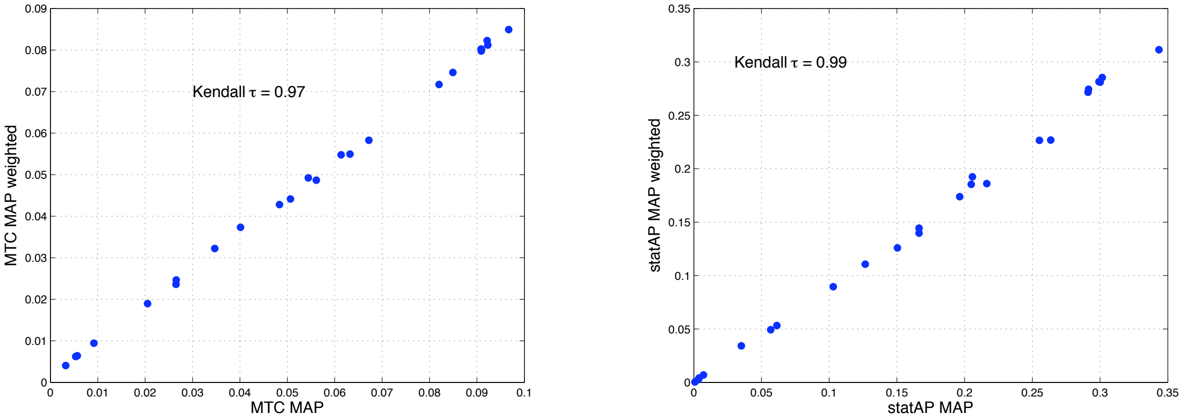

Figure 2: MAP VS weighted MAP for each evaluation method

Judgment targets. We calculated the MAP separately for each set of queries judged to a certain target (8, 16, 32, 64, 128), in order to investigate both stability of the evaluation methodologies and the performance of retrieval systems. The results are presented in Table 6; we provide a detailed analysis in the next section. Due to the differences between the two evaluation technologies, a different number of queries contribute to the MAP for each level, as indicated on the table.

Clearly, the evaluation methods are slightly biased upwards when the number of document judged is very small (i.e. 8); there is a consistent trend (for middle and good systems) to have the MAP decreasing as the number of judgments increases. One of the conclusions of this study is that a reliable evaluation will need at least a dozen documents judged per query.

Query categories. Results in the form of weighted MAP are presented in Table 7. Consistently across systems, *short* and *gov-heavy* queries are the "easiest" (systems have highest performance on these categories). This may be because there are more relevant documents containing query terms that are therefore easier to find, though that may in turn be because short queries are more open to interpretation and thus produce more relevant documents. According to MTC, the hardest category for systems appears to be the *short* and gov-slant queries; statAP performance varies across systems without indicating a clear hardest category of queries.

4.3.1 Other measures

To test whether the other measures do a fair job of ranking systems we correlated system rankings by one measure over a set of queries/judgments to system rankings by another measure over the same set of queries/judgments (Table 8). They tend to correlate very well ($\tau \approx .9$), suggesting that even though our judgments were acquired to rank systems by MAP estimates, they can be used for other measures reliably.

	MTC					statAP				
run	403x8	204x16	102x32	50x64	25x128	271x8	138x16	89x32	43x64	23x128
hedge0	0.003	0.004	0.004	0.005	0.006	0.002	0.000	0.000	0.000	0.000
lsi150stat	0.005	0.006	0.006	0.007	0.008	0.004	0.002	0.002	0.001	0.003
vsmstat	0.005	0.006	0.006	0.007	0.008	0.006	0.000	0.001	0.003	0.010
vsmstat07	0.005	0.006	0.006	0.007	0.009	0.007	0.000	0.001	0.000	0.010
sabmq08a1	0.009	0.009	0.010	0.010	0.009	0.010	0.001	0.007	0.011	0.005
lsi150dyn	0.019	0.022	0.025	0.016	0.011	0.023	0.057	0.044	0.031	0.020
vsmdyn	0.027	0.026	0.031	0.024	0.016	0.054	0.071	0.058	0.046	0.025
000cos	0.027	0.024	0.035	0.017	0.014	0.064	0.058	0.081	0.035	0.033
000tfidfLOG	0.036	0.032	0.038	0.031	0.024	0.119	0.080	0.105	0.103	0.048
000tfidfBM25	0.042	0.036	0.045	0.033	0.030	0.144	0.095	0.145	0.118	0.057
000klabs	0.053	0.044	0.053	0.033	0.032	0.161	0.164	0.139	0.097	0.086
000okapi	0.056	0.045	0.054	0.034	0.031	0.175	0.176	0.156	0.143	0.088
LucDeflt	0.059	0.049	0.059	0.040	0.038	0.213	0.185	0.197	0.164	0.123
sabmq08b1	0.061	0.052	0.061	0.036	0.033	0.173	0.202	0.132	0.121	0.094
indriLowMu08	0.068	0.053	0.063	0.047	0.043	0.246	0.191	0.201	0.195	0.116
mpiimq0801	0.070	0.057	0.067	0.041	0.040	0.219	0.193	0.207	0.188	0.132
neuMSRF	0.074	0.061	0.072	0.047	0.038	0.197	0.215	0.243	0.206	0.111
neustbl	0.091	0.073	0.083	0.056	0.055	0.298	0.254	0.217	0.226	0.164
neumsfilt	0.094	0.075	0.088	0.058	0.057	0.298	0.214	0.213	0.244	0.180
dxrun	0.101	0.081	0.094	0.061	0.061	0.297	0.299	0.277	0.305	0.207
indriQLST08	0.101	0.079	0.093	0.063	0.064	0.297	0.306	0.269	0.300	0.203
txrun	0.101	0.080	0.097	0.061	0.063	0.301	0.324	0.286	0.305	0.226
LucLpTfS	0.102	0.081	0.096	0.064	0.068	0.331	0.249	0.295	0.300	0.239
ind25QLnST08	0.103	0.081	0.093	0.064	0.064	0.304	0.330	0.261	0.295	0.233
indri25DM08	0.108	0.085	0.099	0.065	0.067	0.368	0.358	0.281	0.323	0.254

Table 6: MTC and statAP estimation of MAP separately for sets of queries judged at 8, 16, 32, 64, 128 level. The column header indicates the number of queries judged at a specific level(for example, for first column, 403 queries with 8 documents judged). MTC outputs an estimate for all queries judged, while statAP estimates only queries with at least one judged relevant document; therefore there are less queries per level of judgment reported for statAP.

5 Analysis

The above results are based on a large sample of 784 sparsely-judged queries distributed uniformly over four categories. The next step is to determine the extent to which the number of queries and judgments can be reduced, and how to sample the categories to achieve similar results with less overall effort.

Our aim is to answer the following question: what is the number of queries needed for different levels of relevance incompleteness to guarantee that, when systems are run over this many queries, their MAP scores reflect their actual performance?

5.1 Analysis of variance studies

When systems are run over different sets of queries, MAP scores (and consequently the ranking of these systems) may vary. The amount of variability that occurs in MAP scores (as measured

	MTC				statAP			
	short,heavy	long,heavy	short,slant	long,slant	short,heavy	long,heavy	short,slant	long,slant
run	197queries	194queries	196queries	197queries	155queries	158queries	121queries	130queries
hedge0	0.003	0.003	0.004	0.003	0.000	0.000	0.003	0.000
vsmstat07	0.005	0.005	0.006	0.005	0.006	0.004	0.000	0.000
vsmstat	0.005	0.005	0.006	0.005	0.006	0.004	0.000	0.000
lsi150stat	0.005	0.005	0.006	0.005	0.007	0.001	0.002	0.000
sabmq08a1	0.009	0.009	0.009	0.009	0.003	0.003	0.002	0.008
lsi150dyn	0.022	0.020	0.017	0.020	0.037	0.029	0.014	0.060
vsmdyn	0.031	0.025	0.020	0.024	0.088	0.037	0.026	0.062
000cos	0.044	0.019	0.020	0.016	0.140	0.019	0.047	0.026
000tfidfLOG	0.046	0.032	0.028	0.029	0.117	0.076	0.100	0.108
000tfidfBM25	0.052	0.036	0.033	0.031	0.146	0.099	0.117	0.106
LucDeflt	0.058	0.054	0.040	0.055	0.229	0.164	0.181	0.188
000okapi	0.058	0.045	0.040	0.043	0.227	0.121	0.187	0.137
000klabs	0.062	0.039	0.039	0.038	0.267	0.082	0.167	0.081
indriLowMu08	0.062	0.063	0.046	0.059	0.253	0.216	0.260	0.213
neuMSRF	0.077	0.070	0.051	0.063	0.159	0.335	0.197	0.138
sabmq08b1	0.077	0.052	0.040	0.047	0.248	0.113	0.096	0.109
mpiimq0801	0.084	0.062	0.044	0.051	0.310	0.133	0.214	0.144
neustbl	0.104	0.079	0.059	0.071	0.332	0.317	0.193	0.222
neumsfilt	0.110	0.083	0.063	0.074	0.334	0.325	0.208	0.198
indriQLST08	0.111	0.094	0.064	0.078	0.302	0.255	0.240	0.273
ind25QLnST08	0.113	0.095	0.066	0.082	0.294	0.328	0.260	0.264
dxrun	0.116	0.092	0.067	0.077	0.299	0.326	0.300	0.281
LucLpTfS	0.116	0.092	0.070	0.082	0.306	0.289	0.237	0.262
txrun	0.118	0.090	0.068	0.077	0.315	0.327	0.292	0.288
indri25DM08	0.120	0.098	0.069	0.085	0.369	0.308	0.288	0.292

Table 7: MTC and statAP estimation of MAP (weighted w.r.t. numbers of judgments per query) separately for each category set of queries.

by variance) across all sets of queries and all systems can be decomposed into three components: (a) variance due to actual performance differences among systems — *system variance*, (b) variance due to the relative difficulty of a particular set of queries — *query set variance*, and (c) variance due to the fact that different systems consider different set of queries hard (or easy) — *system-query set interaction variance*. The variance due to other sources of variability, such as sampling of documents, is confounded with the later variance component, i.e. variance due to *system-query set interaction*.

Ideally, one would like the total variance in MAP scores to be due to the actual performance differences between systems, as opposed to the other two sources of variance. In such a case, having the systems run over different sets of queries would result into each system obtaining identical MAP scores over all sets of queries, and thus MAP scores over a single set of queries would be 100% reliable in evaluating the quality of the systems. The percentage of variance attributed to the *system* effect is a function of the size of the query set.

Note that among the three variance components, only the variance due to *system* and *system-query set interaction* affects the *ranking* of systems — it is these two components that can alter the relative differences among MAP scores, while the *query set* variance will affect all systems equally, reflecting the overall difficulty of the set of queries.

In practice, retrieval systems are run over a single given set of queries. The decomposition of the total MAP variance into the aforementioned components in this case can be realized by fitting

	MTC				statAP			
run	Rprec	prec@10	prec@30	prec@100	Rprec	prec@10	prec@30	prec@100
hedge0	0.048	0.022	0.030	0.055	0.000	0.001	0.001	0.000
vsmstat07	0.059	0.046	0.053	0.070	0.016	0.040	0.024	0.013
vsmstat	0.059	0.048	0.054	0.069	0.017	0.045	0.027	0.014
lsi150stat	0.060	0.050	0.055	0.071	0.009	0.086	0.041	0.014
sabmq08a1	0.071	0.079	0.080	0.085	0.012	0.034	0.038	0.044
000cos	0.104	0.127	0.138	0.128	0.114	0.094	0.155	0.109
lsi150dyn	0.091	0.138	0.127	0.110	0.067	0.135	0.136	0.123
vsmdyn	0.103	0.163	0.146	0.125	0.093	0.136	0.148	0.126
000tfidfLOG	0.115	0.206	0.175	0.141	0.166	0.309	0.236	0.169
000tfidfBM25	0.122	0.222	0.189	0.150	0.194	0.294	0.245	0.190
000klabs	0.130	0.227	0.202	0.160	0.201	0.249	0.264	0.178
000okapi	0.133	0.243	0.212	0.163	0.219	0.281	0.287	0.205
LucDeflt	0.143	0.255	0.221	0.176	0.244	0.331	0.278	0.223
indriLowMu08	0.146	0.280	0.241	0.181	0.271	0.300	0.290	0.247
sabmq08b1	0.133	0.282	0.228	0.165	0.202	0.391	0.314	0.281
mpiimq0801	0.139	0.316	0.248	0.171	0.245	0.384	0.355	0.235
neuMSRF	0.146	0.328	0.263	0.181	0.261	0.440	0.398	0.261
neustbl	0.161	0.359	0.289	0.201	0.302	0.417	0.367	0.282
dxrun	0.170	0.373	0.309	0.212	0.366	0.399	0.398	0.315
indriQLST08	0.170	0.379	0.309	0.212	0.349	0.431	0.395	0.306
txrun	0.169	0.382	0.310	0.211	0.373	0.454	0.406	0.311
neumsfilt	0.161	0.385	0.303	0.200	0.316	0.444	0.402	0.291
ind25QLnST08	0.171	0.389	0.311	0.213	0.355	0.448	0.388	0.296
indri25DM08	0.174	0.398	0.325	0.218	0.390	0.475	0.403	0.341
LucLpTfS	0.172	0.407	0.322	0.215	0.359	0.506	0.408	0.317

Table 8: MTC and statAP estimation of other measures. The numbers are weighted averages over queries. Ordered by MTC estimate of prec@10.

an Analysis of Variance (ANOVA) model into AP scores [BOZ99, BL07, Bre01].

5.1.1 Stability of wMAP scores and induced rankings

We ran two separate variance decomposition studies; one over the MAP scores estimated by the MTC method and one over the MAP scores estimated by the statAP method. In both cases systems were run over the same set of 784 queries[2] and each one of the methods utilized all available judgments per query in the estimation of MAP scores.

As mentioned before, first, MAP scores were computed over each of the judgment targets separately (MAP_j), and then averaged to produce the final MAP scores (wMAP). The variance in wMAP is a function of the variance of the MAP within each query class and the covariance of the MAP values among query classes. Thus, instead of fitting a single ANOVA model in AP values over all queries [BOZ99, BL07], we used a Multivariate Analysis of Variance (MANOVA) [Bre01].

[2] Note that statAP does not report scores for queries with no relevant document found. Therefore, studies for statAP were only on the 564 queries for which statAP returned scores.

The variance of MAP within each query class was decomposed into the aforementioned variance component, while the covariance of the MAPs among the query classes was solely attributed to *system* effects, since the query classes are disjoint.

For both studies, we report (a) the stability levels of the wMAPs (the ratio of the variance due to system and the total variance) and (b) the stability levels of the systems rankings (the ratio of the variance due to system and the variance components that affect the relative MAP scores, i.e. the ranking of systems), both as a function of the total number of queries in the query set. The results of the two studies are illustrated in Figure 3.

The solid lines correspond to stability levels of wMAP values which express how fast (in terms of number of queries) we reach stable wMAP values over different sets of queries of the same size. As the figure indicates, statAP reaches a stability level of 0.95 with a set of 129 queries, while MTC reaches the same level with 204 queries (not observed in the figure). [3]

The dashed lines correspond to stability levels of systems ranking which expresses how fast (in terms of number of queries) we reach stable system rankings. MTC reaches a stability level of 0.95 with a set of 83 queries, while statAP reaches the same level with 102 queries.

These results support the claims that the statAP method, by design, aims to estimate the actual MAP scores of the systems, while the MTC method, by design, aims to infer the proper ranking of systems.

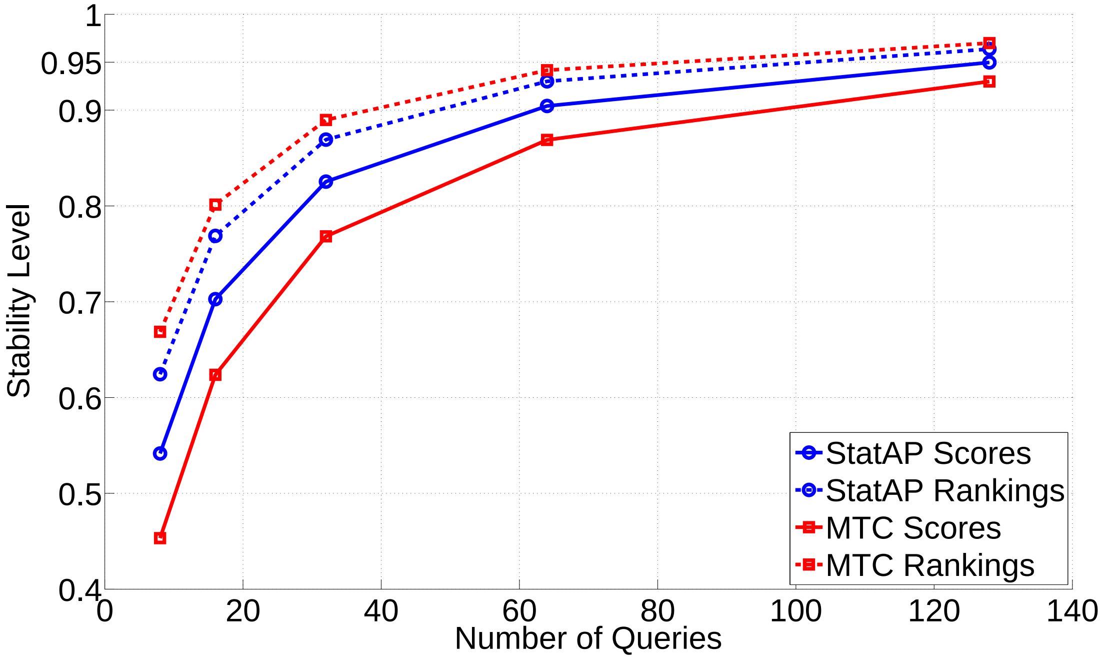

Figure 3: Stability level of MAP scores and induced ranking of systems for statAP and MTC as a function of the number of queries.

[3] We have observed in our experiments that a stability of 0.95 leads to a Kendall's tau of approximately 0.9.

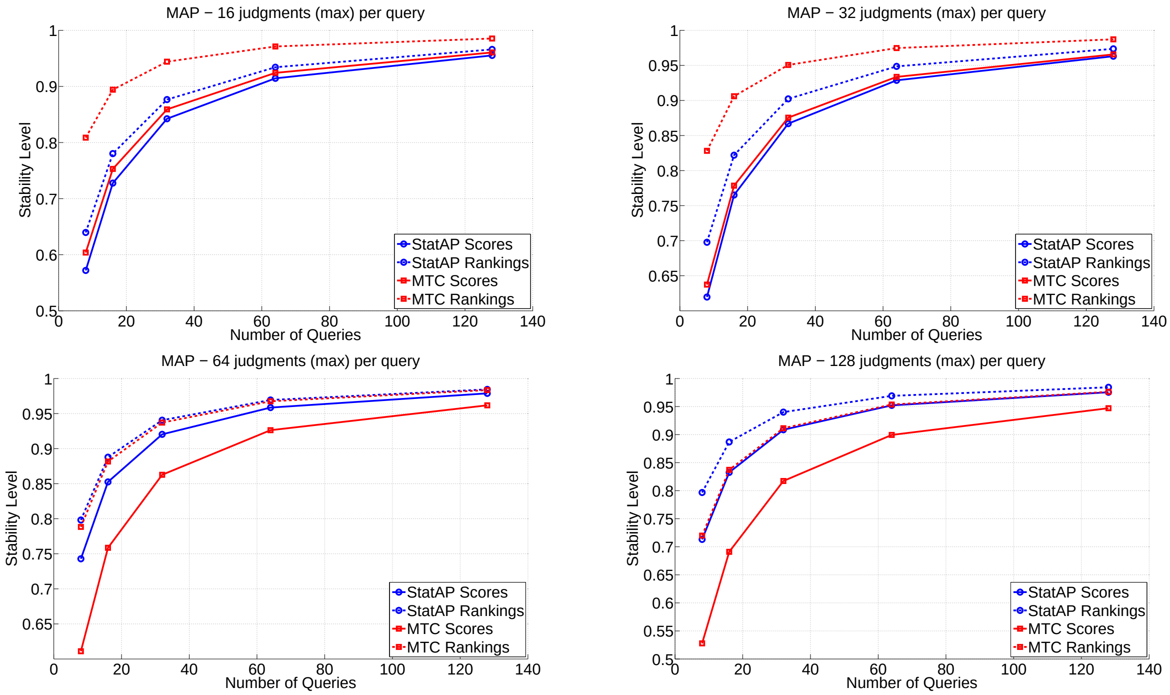

Figure 4: Stability levels of MAP scores and induced ranking for statAP and MTC as a function of the number of queries for different levels of relevance incompleteness.

5.1.2 Stability for different levels of relevance incompleteness

To illustrate how stable the MAPs returned by the two methods are with respect to different levels of relevance incompleteness, we ran ANOVA studies for each one of the query classes separately. Figure 4 demonstrate the stability levels for both methods when 16, 32, 64, and 128 judgments are available, respectively.

MTC leads to both more stable MAPs and induced rankings than statAP when 16 or 32 relevance judgments are available per query, while the opposite is true when 64 or 128 relevance judgments are available.

Note that the stability of the MAP scores returned by statAP degrades with the relevance incompleteness, as expected. On the other hand, the opposite is true for MTC. For the estimation of MAP scores, MTC is employing a prior distribution of relevance which is calculated by combining information from all queries, which violates the query independence assumption ANOVA makes. The fewer the relevance judgments, the larger the weight on the prior distribution, and thus the more the assumption is violated. Consequently, MAP scores seem to be more stable than they should.

5.1.3 Stability for different query categories

Furthermore, to illustrate how stable the wMAP values returned by the two methods are with respect to the different query categories (i.e. short, long, govslant and govheavy), we also ran MANOVA studies for each one of the query categories separately in the same manner as in Subsection 5.1.1. Figure 5 demonstrate the stability levels for both methods.

Both methods lead faster to stable results (both wMAP values and induced rankings) for long

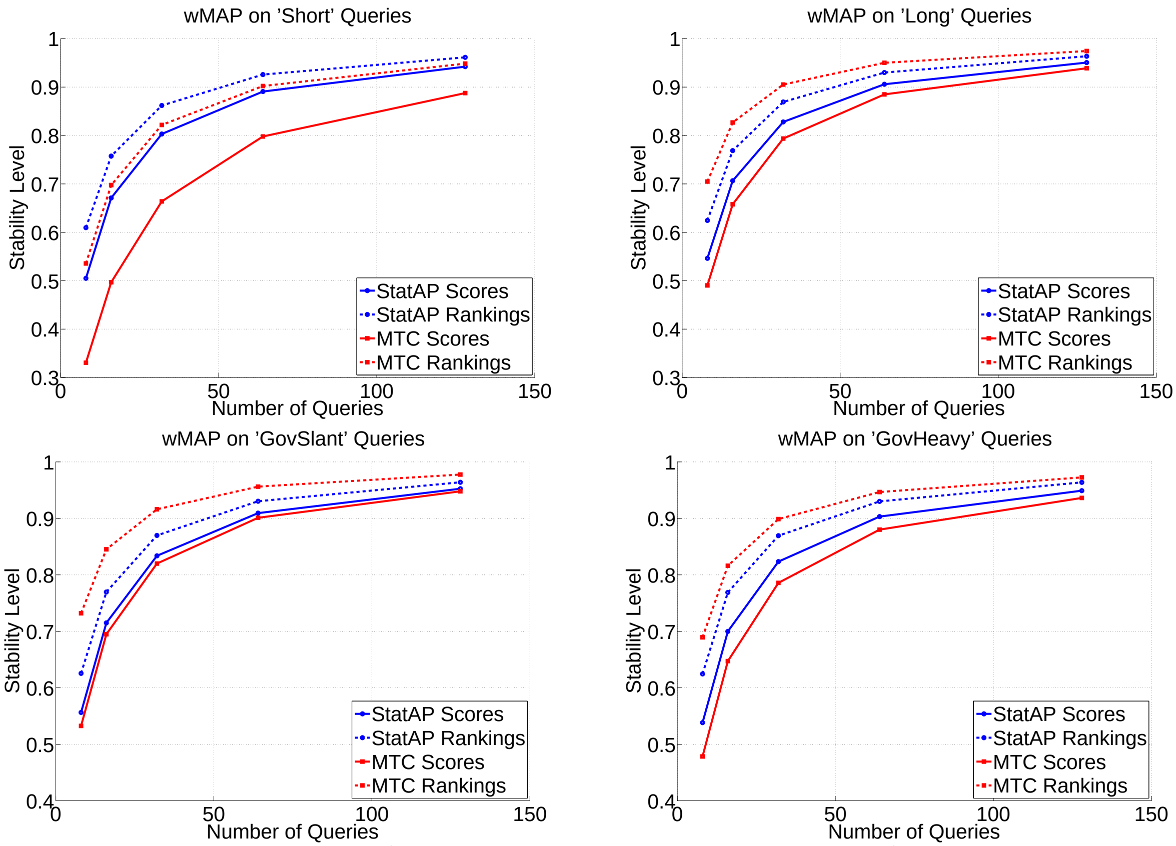

Figure 5: Stability levels of wMAP scores and induced ranking for statAP and MTC for different query categories.

than for short queries (top plots) and for govslant than for govheavy queries (bottom plots). Furthermore, the stability of MTC results appears to be affected more from the different query categories than the stability of statAP results.

6 Conclusion

Many queries; categories. We put in practice two recently developed evaluation techniques that, unlike standard evaluation, scale easily and allow many more experiments and analyses. We experimented with 25 submitted systems over 10000 natural queries, evaluating 784 of them with only about 15000 judgments.

We investigated system performance over pre-assigned query categories. There is some evidence that over-sampling some types of queries may result in cheaper (if not substantially more efficient) evaluation. We have presented more such evidence in a separate work [CPK+09].

Evaluation stability. The setup also allowed an analysis of evaluation stability with fewer judgments or queries. Using ANOVA, we concluded that MTC (ranking optimized) needs about 83 queries with approximately 1700 total judgments for a reliable ranking, while statAP (score optimized) needs about 129 queries with approximately 2650 total judgments for a reliable estimate of MAP.

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 HR0011-06-C-0023, in part by the National Science Foundation (NSF) under grant numbers IIS-0533625 and IIS-0534482, and in part by Microsoft Live Labs. Any opinions, findings and conclusions or recommendations expressed in this material are the authors and do not necessarily reflect those of the sponsor.

References

[ACA+07] James Allan, Ben Carterette, Javed A. Aslam, Virgil Pavlu, Blagovest Dachev, and Evangelos Kanoulas. Overview of the TREC 2007 Million Query Track. In *Proceedings of TREC*, 2007.

[AP] Javed A. Aslam and Virgil Pavlu. A practical sampling strategy for efficient retrieval evaluation, technical report.

[AP07] Javed A. Aslam and Virgil Pavlu. A practical sampling strategy for efficient retrieval evaluation. Working draft available at the following URL: http://www.ccs.neu.edu/home/jaa/papers/drafts/statAP.html, May 2007.

[APY06] Javed A. Aslam, Virgil Pavlu, and Emine Yilmaz. A statistical method for system evaluation using incomplete judgments. In *Proceedings of SIGIR*, pages 541–548, 2006.

[BH83] K. R. W. Brewer and M Hanif. *Sampling With Unequal Probabilities.* Springer, New York, 1983.

[BL07] David Bodoff and Pu Li. Test theory for assessing ir test collection. In *Proceedings of SIGIR*, pages 367–374, 2007.

[BOZ99] David Banks, Paul Over, and Nien-Fan Zhang. Blind men and elephants: Six approaches to trec data. *Inf. Retr.*, 1(1-2):7–34, 1999.

[Bre01] Robert L. Brennan. *Generalizability Theory.* Springer-Verlag, New York, 2001.

[CAS06] Ben Carterette, James Allan, and Ramesh K. Sitaraman. Minimal test collections for retrieval evaluation. In *Proceedings of SIGIR*, pages 268–275, 2006.

[CPK+08] Ben Carterette, Virgil Pavlu, Evangelos Kanoulas, James Allan, and Javed A. Aslam. Evaluation over thousands of queries. In *Proceedings of SIGIR*, pages 651–658, 2008.

[CPK+09] Ben Carterette, Virgil Pavlu, Evangelos Kanoulas, Javed A. Aslam, and James Allan. If i had a million queries. In *Proceedings of ECIR*, pages 288–300, 2009.

[Ste] W. L. Stevens. Sampling without replacement with probability proportional to size. *Journal of the Royal Statistical Society. Series B (Methodological), Vol. 20, No. 2. (1958), pp. 393-397.*

[Tho92] Steven K. Thompson. *Sampling.* Wiley Series in Probability and Mathematical Statistics, 1992.

[YA06] Emine Yilmaz and Javed A. Aslam. Estimating average precision with incomplete and imperfect judgments. In *Proceedings of CIKM*, pages 102–111, 2006.