

TREC 2017 Common Core Track Overview

James Allan, Donna Harman, Evangelos Kanoulas,
Dan Li, Christophe Van Gysel, Ellen Vorhees*

1 Introduction

The primary goal of the TREC Common Core track is three-fold: (a) bring the information retrieval community back into a traditional ad-hoc search task; (b) attract a diverse set of participating runs and build a new test collections using more recently created documents; (c) establish a (new) test collection construction methodology that avoids the pitfalls of depth-k pooling. A number of side-goals are also set, including studying the shortcomings of test collections constructed in the past; experimenting with new ideas for constructing test collections; expand test collections by new participant tasks (ad-hoc/interactive), new relevance judgments (binary/multilevel), new pooling methods, new assessment resources (NIST / crowd-sourcing) and new retrieval systems contributing documents (manual/neural/strong baselines).

2 Task Description

The participants task of the track is a traditional ad-hoc retrieval task. The organizers of the track provided participants with the title/description/narrative of TREC topics and allowed participating sites to run the experiment as they wish as long as they contribute a ranked list of documents as an output.

3 Test Collection

3.1 Corpus

The corpus used is the New York Times Annotated Corpus (<https://catalog.ldc.upenn.edu/ldc2008t19>). The corpus contains over 1.8 million articles written and published by the New York Times between January 1, 1987 and June 19, 2007 with article metadata provided by the New York Times Newsroom, the New York Times Indexing Service and the online production staff at nytimes.com. The text in this corpus is formatted in News Industry Text Format (NITF) developed by the International Press Telecommunications Council, an independent association of news agencies and publishers. NITF is an XML specification that provides a standardized representation for the content and structure of discrete news articles. NITF encompasses structural markup such as bylines, headlines and paragraphs.

3.2 Topics

The topics provided are an updated version of the TREC 2004 Robust track. Most of the topics remain the same as in the 2004 Robust track, but some were revised to reflect the time past (e.g. some descriptions and narratives were turned into historical topics, while others were brought up to date).¹

* Authors are in alphabetical order.

¹Participants have the opportunity to train their systems over the TREC 2004 Robust track (which, to some extent, alters the participants task to a routing task, i.e. fixed queries and updated corpus).

3.3 NIST Relevance Judgments

Two of the goals of the track were to 1) build a new ad hoc test collection by 2) exploring new collection construction methodologies. The original TREC ad hoc collections were built using depth- k pooling, which can create good collections but does so at a relatively high cost. The desiderata for the new methodology is that it build reusable collections (i.e., a collection that is fair even to systems that did not contribute to the building process) at minimal cost.

Multi-arm bandit methods have been demonstrated to be a cost-effective approach to building quality collections in simulations with existing TREC collections [Losada et al., 2016]. The general form of a bandit method for a single topic is as follows. A document selector process starts with an empty set of judgments and the ranked list of documents from each of a set of contributing runs. The selector picks the first document in the ranking of some run and receives a (binary) relevance judgment for that document. Based on the relevance judgments obtained so far and the ranks at which those documents appear in the runs, the selector chooses the next document to be judged. Bandit methods differ in how they choose this next document—whether to choose the subsequent document from the current run or to choose a different run. However, all of the methods are constrained to select the first (closest to rank 1) unjudged document from whichever run has been selected at each step. The idea behind the bandit methods is to focus assessing resources on those runs that continue to contribute relevant documents.

A particular bandit method, Losada et al. [2016]’s “MaxMean” or “MM” (stationary) method, was used in the Common Core track. In this method, each run is assigned a weight after each judgment is obtained, and the first unjudged document in the run with the highest weight is selected as the next document to be judged. The weight of a run is equal to $\frac{1+\text{num-relevant-retrieved}}{2+\text{num-judged}}$. Whenever a document is judged, the number of relevant documents retrieved and the number of documents judged statistics are updated for all runs that retrieved that document, not just the run from which it was selected. For these purposes, a run has “retrieved” a document if the document appears in the top X ranks; we used $X = 100$. A run is randomly chosen from among those runs with the highest weight when there are ties.

Bandit methods are *dynamic methods* in that they require a relevance judgment for the current document to select another document to be judged. The logistical challenges in implementing dynamic methods for actual use (as opposed to simulation) is one reason why NIST had not used them to build collections in the past. But there are two other concerns as well: bias against systems that do not have good early precision and bias in assessor judgments since assessors see documents in rank order.

The assessor bias concern remains; we do not know of any way to control for assessor bias when using a dynamic method nor a way to gauge its effect (if any). Assessors were not told they would be seeing documents in any particular order, but did quickly figure out that they were seeing the best documents first. However, the results of a series of experiments on past collections detailed below suggest that the bandit methods are as fair to all runs participating in the judgment process as other collection-building methods. Further experiments demonstrated the way forward through the logistical challenges.

These experiments were carried out on two collections: the TREC-8 ad hoc collection and the collection built from the TREC 2005 Robust and HARD track submissions. The TREC-8 collection is among the most-studied IR test collections and is a very solid collections. It has more than 80,000 judgments across 50 topics and was built using depth-100 pooling over a strong set of runs. The TREC 2005 collection has just under 38,000 judgments, also across 50 topics, and was constructed using depth-55 pooling. This collection was the collection that first demonstrated the limits of pooling [Buckley et al., 2007], in particular that a sufficient pooling depth depends on collection size and so must grow as collection size grows. Among the runs used to construct the collection is a run that contributed a large percentage of unique relevant documents. “Leave one out” tests show significant bias against this run when its uniquely retrieved relevant are omitted from the existing judgment sets.

3.3.1 Bias against “slow” runs

By design, bandit methods concentrate assessing resources on runs that have been demonstrated to deliver relevant documents. Since the bandit methods start from the first ranks of a set of runs and work their way down the ranked lists, the fear is that bandit methods could not find pockets of relevant documents in runs whose top documents were not relevant.

TREC-8				TREC 2005			
k	Pool	MM	Inferred	k	Pool	MM	Inferred
10	.914	.963	.928	10	.829	.839	.847
20	.945	.982	.965	20	.886	.876	.914
30	.964	.989	.975	30	.902	.905	.927
40	.976	.994	.982	55	.931	.931	.931
50	.985	.997	.987				
60	.990	.998	.989				
70	.994	.999	.991				
80	.996	.999	.994				
90	.998	1.000	.996				

Table 1: Kendall τ values between rankings produced using original qrels and rankings produced using test qrels for depth- k pooling (Pool), MaxMean bandit sampling (MM) and inferred measures sampling (Inferred). The size of the depth- k pool determines the budgets for the other two methods, and k is constrained by the original collection’s pool depth.

To test whether the concern was well-founded, we ran a series of simulations of different collection-building techniques using the two existing collections and compared the quality of the resulting judgments sets (called *qrels*). The existing full-collection qrels (“original qrels”) is taken as ground truth. The quality of qrels formed by other methods (“test qrels”) was measured by how closely a ranking of runs produced by the test qrels matched the ranking produced by the original qrels. The similarity of two rankings was computed as the Kendall τ score between them.

The experiments compared qrels built using depth- k pooling, qrels built through sampling in support of inferred measures [Yilmaz et al., 2008], and qrels built using the MaxMean bandit method for a range of values of k . To control for the total number of documents being judged, the depth- k pools were first constructed, and then the bandit method was given a budget equal to the size of the pool for that topic. The total number of documents in the pools over all 50 topics was given to the inferred-sampling method as its budget. This is somewhat unfair to the inferred-sampling method since the pool and bandit methods had variable budgets per topic while the inferred-sampling method had an equal budget per topic. The inferred-sampling method always used the same two-strata sampling strategy of judging all of the documents in the top 10 ranks (i.e., top-10 pools) plus a sample of documents in ranks 11-100 at the rate determined by maximizing the total number of judgments obtained while staying within the overall budget.

Table 3.3.1 shows the τ values obtained using system rankings based on MAP (or infAP) scores for the different test qrels produced at different depths (k). The set of runs being ranked was the set of all runs (both judged and unjudged) that were submitted to the ad hoc task for TREC-8 and to the Robust track for TREC 2005. The τ values shown for the inferred-sampling method are means over 10 different trials where a given trial is based on an independent sample of documents to be judged. (The original qrels contain documents from runs other than those used in the simulations. So the depth-55 pools for the TREC 2005 collection in the simulations are not identical to the original qrels.)

The test qrels are all able to rank runs much the same as the original qrels. For the TREC-8 collection, the MaxMean method is somewhat better than the inferred-sampling method which is better than depth- k pooling, but even depth-10 pools rank systems very similarly as the original depth-100 pools. There is more of a difference for the TREC 2005 collection, and here the inferred-sampling method is somewhat better than the MaxMean method. But both methods still create qrels that rank systems very similarly as the original, especially for pool depths that are likely to be used in practice, demonstrating that fears of bias against unjudged runs for the MaxMean method are unfounded for these collections. This could be because the set of runs submitted to these tracks just did not exhibit this behavior, or because the behavior is truly rare, or because even though some runs do have this behavior other runs compensate for them, or since evaluation measures concentrate on top ranks system comparisons are unaffected even when the behavior is present, or other reasons and combinations of reasons. In any event, there is no discernible bias against judged runs by the MaxMean method in these tests.

3.3.2 Logistical challenges

Using the MaxMean method to actually build a new collection presents logistical challenges not encountered in simulations. For all dynamic methods, a judgment once obtained is assumed to be immutable and has instantaneous effect on which documents are subsequently selected. But human assessors need a “burn-in” period to learn the nuances of a topic. During a burn-in period, the assessor may make changes to past judgments to improve consistency with their current understanding of the topic. To accommodate a burn-in period, the track did not use a ‘pure’ MaxMean method, but instead first created depth-10 pools that assessors judged normally, and then used the judged pool as the initial set of judgments for the MaxMean bandit process. In addition to providing a burn-in period for the assessor, the top-10 pools allowed the track to guarantee that all teams would at least get judgments for the top ten documents in their judged runs. The top-10 pools also provided a means for setting per-topic assessment budgets, which was the second logistical challenge.

Simulations avoid the problem of deciding how to allocate an overall budget among individual topics by simply changing the order in which documents from some larger pool are encountered. But when building the collection from scratch, there needs to be some concrete decision as to when the dynamic method will stop. A simple technique is to give each topic an equal share of the overall budget, but this is known to be suboptimal. Different topics need more or fewer judgments depending on run overlap and total number of relevant documents, and a final qrels can be substantially improved if topics needing the larger number of judgments receive larger budgets. Another approach is to apply a bandit method to the question of which topic to judge next and then select the next document for that topic. In simulations, this dual-selection process does indeed generate very good qrels for a given overall budget, but it has a fatal flaw. A given topic needs to be judged by a single assessor and assessors should concentrate on one topic at a time because constant context switches both increase the time it takes to make a judgment and decreases the quality of those judgments. At NIST, multiple assessors work on their own set of topics at different times and at different rates, but the budget for one topic depends on the budget given to all the rest. So NIST decided to set the budgets for all topics at one time at the start of the dynamic process.

We used the information from the judged top-10 pools to create the individual topic budgets. For past collections, we have not only the judgments themselves, but also the ranks at which each relevant document was retrieved in a judged run and the documents in the top-10 pools. Using the ranks at which the relevant documents appear, we can determine the minimal number of documents to select from each run for each topic, subject to the constraint that all selections must start at the top of a run’s document ranking, to get the maximum number of relevant documents from the original qrels that are obtainable given the overall budget. We call this an optimal qrels—optimal for any bandit method with the given budget, though likely inferior to the original qrels. There may be more than one optimal qrels for a budget, and any given optimal qrels may have multiple different selections from runs to obtain it. Using the minimal number of judgments per topic required to obtain an optimal qrels, we fit a model to predict how many judgments a topic should receive as a function of three statistics from top-10 pools: the overall pool size, the number of relevant documents in the pool, and the number of nonrelevant documents in the pool. This process demonstrated $topfactor_t = \frac{|pool_t|}{\sqrt{|nonrel-in-pool_t|}}$ to be a simple, effective factor for selecting topic t ’s budget. For the Common Core track’s collection, once the top-10 pools were judged, we computed the $topfactor$ for each topic and summed them to produce a normalizing factor, T . The residual budget remaining after pool judging was allocated to the individual topics by assigning $(\frac{topfactor_t}{T} \times residual)$ judgments as topic t ’s budget.

A third logistical challenge this year was introduced by having two separate run submission deadlines for the track separated by approximately one month. Two-thirds of the NIST assessing time was scheduled to occur during the month between deadlines, with the remaining third following the second deadline. Thus runs submitted at the earlier deadline could potentially receive many more judgments than runs submitted at the later deadline, even if the earlier runs were less effective. Yet another set of experiments was run on the TREC-8 collection to gauge the effect of unbalanced assessment time for different run sets on the quality of the final qrels.

In these experiments, an overall assessment budget B was picked and the TREC-8 judged runs were randomly assigned to either the “early” set or the “later” set. Top-10 pools were created from the runs in the early set and individual topic budgets were assigned as described above using a budget of $2/3 \times B$. The MaxMean method was run until those individual topic budgets were exhausted. The top-10 pools over the combined set of runs (early and later runs) was then produced and individual topic budgets produced using these pools and budget B . The MaxMean

process was given the combined runs' top-10 pools and budgets, and began anew using the combined run set. When a document selected to be judged was already judged (because it was in the pool or in the early runs' bandit processing) the statistics for the runs were updated normally but the budget was not decreased. The MaxMean continued in this vein until the topic budgets were exhausted. The entire process was repeated 50 times using 50 different independent splits of runs into early and later runs.

A test qrels that resulted from this two-stage process was compared to the original qrels by computing the root mean square error (RMSE) of the number of relevant documents for each topic in each qrels. Given the variance in the number of relevant documents across topics, RMSE can be misleading since a qrels can have a relatively small RMSE if it contains many relevant documents for large topics even if it contains no relevant documents for small topics. Manual inspection of the results suggests that this did not happen, however. When using a budget of $B = 36,000$, the RMSE values ranged from 9.98–19.63 indicating at most a modest impact of the the split of runs on final qrels quality. The larger RMSE's occurred when the early set contained few (less than 10) runs. This year's Common Core track received 41 runs by the early deadline and 34 runs at the later deadline, so the impact of the two-stage process should be negligible.

3.3.3 Implementation

As noted, the track received a total of 75 runs from 14 participant teams and a pseudo-team called "BASELINE". Teams could submit up to 10 runs total and at most 3 runs by the early deadline. All 41 runs that were received at the first deadline contributed to the judging process in the first stage. At most 5 runs per team contributed to the judgment process in the second stage, resulting in a total of 55 judged runs. Several runs that were submitted at the early deadline were mistakenly omitted from the second round of judging when they were replaced by new runs from the same team. This means the qrels file released for the track contains about 50 relevant documents that were not retrieved by any of the runs that are officially designated as judged runs (where "retrieved" means in the top 100 ranks).

The server implementing the MaxMean method was developed by Aldo Lipani when he was visiting NIST. This version of the server accepts an arbitrary set of judgments as an initial qrels and adds to that qrels after receiving a judgment. Any document selected to be judged that is already in the qrels is not returned to the assessor for a second time. This version also enforces that top- k pools are judged for all runs that are participating in the process, and limits the depth that will be mined from a single run to no more than X . For the Common Core track, we used $k = 10$ and $X = 100$.

NIST estimated the total number of assessment that could be obtained in the time allocated for Common Core track judging to be between 30,000–35,000, so the budget set in the first stage was 22,000. However, assessing went more slowly than estimated, which was further complicated by the fact that the use of MaxMean meant that there was much less flexibility to move topics between assessors and that faster assessors had to wait for slower assessors to finish pool judgments so the bandit budgets could be set. The result was only about 15,000 judgments were made in the first stage, and while all 50 topics had at least some judgments, the number of judgments per topic reflected only whether an assessor had gotten to the topic. NIST was able to obtain more assessing time for the track for the second stage. Individual topic budgets were set based on estimates of the number of nonrelevant there would be in the combined top-10 pools using the first stage judgments to make the estimates, and then allocated as described above. With the additional time, the target number of 30,000 judgments was reached. Nonetheless, the total number of judgments for a topic in the final qrels has been skewed by the imbalance in the first stage, and so the final count does not accurately reflect the intended relative budgets for topics as predicted by *topfactor*.

In addition to the top-10 pools, the static judgment sets that were assessed at the beginning of the second stage also contained documents from an inferred-measure sample. For this sample, the two strata were again a top-10 pool (conveniently already judged) and a 20% sample of the documents retrieved in ranks 11–75. This sample was added so that a direct comparison can be made between the MaxMean and inferred measure approaches on this new collection (this analysis has not yet been done). This is another reason why the qrels released for the track do not reflect a pure MaxMean sampling approach.

3.4 Crowd-sourced Assessments

The Information and Language Processing Systems group at the University of Amsterdam, the Center for Intelligent Information Retrieval at the University of Massachusetts at Amherst, and the Web & Media group at the VU University Amsterdam, funded a crowd-sourcing experiment with the goal of amending the NIST judgments by providing judgments both for the 50 NIST queries, and for the remaining 200 Robust track queries.

At the time of the writing of this paper a pilot crowd-sourcing experiment has been conducted to fine-tune the parameters of actual experiment to follow. The micro-task for the pilot experiment is simple: the title and the description of a topic is provided to a crowd-worker, along with a NYT article, with the crowd-worker asked to tag the document as either highly relevant, relevant or not relevant. For the pilot experiment a total of 300 topic-article pairs were selected. Articles were split into 5 bins on the basis of their length, and 60 articles were chosen from each bin; different payments were given to crowd-workers for the different bins with bin 1 and 2 paying 0.02\$, bin 3 0.03\$, and bin 4 and 5 0.04\$. Quality metrics computed by the CrowdTruth framework (<http://crowdtruth.org/>) were positive, hence the full experiment is ready to start.

3.5 Evaluation Measures

Retrieval Effectiveness: The retrieval effectiveness was measured by traditional evaluation metrics produced by trec_eval. For the track three measures were selected as official measures, Average Precision, Precision@10, and nDCG. Further, the effectiveness of the systems biased the document selection method both in NIST and in Crowd-sourced judgments.

Unique Documents Contribution: The number of unique documents contributed to the pool was not incorporated in any evaluation metric, but rather measured in a post-hoc analysis.

4 Submitted Runs

As noted earlier the track received a total of 75 runs from 14 teams, out of which 55 runs contributed to the pool. Runs varied between automatic and manual, making use of the Robust 2004 track judgments or not.

5 Evaluation

Table 2 demonstrated the mean performance of the 75 submitted runs over the 50 NIST topics, while Figure 5 also shows the varying performance as measured by MAP, for each one of the submitted runs.

Participating Run	MAP	NDCG	PC@10	Participating Run	MAP	NDCG	PC@10
ICT17ZCJL01	0.18	0.404	0.442	ims_wcs_p10	0.267	0.513	0.598
ICT17ZCJL02	0.188	0.413	0.448	ims_wcs_rbp	0.268	0.513	0.604
ICT17ZCJL03	0.148	0.371	0.4	ims_wcs_recall	0.259	0.509	0.586
ICT17ZCJL05	0.141	0.354	0.372	ims_wcs_rprec	0.265	0.512	0.588
ICT17ZCJL06	0.159	0.375	0.42	ims_wcs_twist	0.265	0.512	0.592
ICT17ZCJL07	0.178	0.402	0.458	mpiik10e105akDT	0.166	0.314	0.55
IpsUvABoir	0.286	0.515	0.57	mpiik10e111akDT	0.156	0.307	0.518
IpsUvANvsm	0.126	0.332	0.322	mpiik15e74akDT	0.164	0.312	0.544
IpsUvAQImNvsm	0.171	0.412	0.418	sab17coreA	0.272	0.53	0.614
MRGrandrel	0.319	0.578	0.566	sab17coreE1	0.276	0.515	0.646
MRGrankall	0.354	0.599	0.642	sab17coreO1	0.298	0.537	0.614
MRGrankrel	0.361	0.604	0.65	sabchmergeav45	0.405	0.673	0.748
RMITFDMQEA1	0.212	0.441	0.516	sabchoiceaqv45	0.343	0.607	0.718
RMITRBCUQVT5M1	0.341	0.6	0.712	sabchoicev45	0.395	0.661	0.74
RMITUQVBestM2	0.268	0.523	0.62	sabmerge50aqv45	0.416	0.676	0.762
UDelInfoEXPint	0.284	0.525	0.524	sabopt50av45	0.351	0.619	0.724
UDelInfoLOGext	0.252	0.492	0.53	sabopt50v45	0.376	0.645	0.696
UDelInfoLOGint	0.298	0.539	0.55	tgncorpBASE	0.255	0.51	0.526
UWatMDS_AFuse	0.434	0.685	0.706	tgncorpBOOST	0.276	0.535	0.594
UWatMDS_AUnion	0.397	0.653	0.654	udelIndri	0.232	0.483	0.526
UWatMDS_AWgtd	0.398	0.658	0.676	udelIndriB	0.232	0.483	0.526
UWatMDS_BFuse	0.441	0.694	0.73	umass_baselnrm	0.275	0.499	0.572
UWatMDS_BUnion	0.384	0.64	0.638	umass_baselnsdm	0.228	0.451	0.528
UWatMDS_BWgtd	0.416	0.663	0.738	umass_direlm	0.245	0.481	0.456
UWatMDS_HT10	0.408	0.666	0.462	umass_direlmnvs	0.23	0.473	0.466
UWatMDS_TARsv1	0.462	0.699	0.834	umass_diremart	0.251	0.498	0.462
UWatMDS_TARsv2	0.438	0.678	0.81	umass_emb1	0.235	0.476	0.53
UWatMDS_ustudy	0.32	0.566	0.654	umass_erm	0.289	0.517	0.57
WCrobust04	0.371	0.637	0.646	umass_letor_lm	0.259	0.493	0.538
WCrobust0405	0.428	0.696	0.75	umass_letor_lmn	0.244	0.485	0.492
WCrobust04W	0.366	0.63	0.658	umass_letor_m	0.274	0.514	0.512
ims_bm25_td	0.243	0.487	0.57	umass_maxpas150	0.213	0.438	0.484
ims_cmbsum	0.252	0.503	0.572	umass_maxpas50	0.19	0.411	0.426
ims_dfrinl2_td	0.243	0.485	0.574	webis_baseline	0.066	0.148	0.366
ims_wcmbsum_ap	0.268	0.515	0.59	webis_baseline2	0.066	0.148	0.366
ims_wcs_ap_uf	0.019	0.125	0.016	webis_reranked	0.053	0.135	0.28
ims_wcs_err	0.267	0.514	0.6	webis_reranked2	0.053	0.135	0.28
ims_wcs_ndcg	0.261	0.51	0.584				

Table 2: Evaluation scores over the 50 NIST topics.

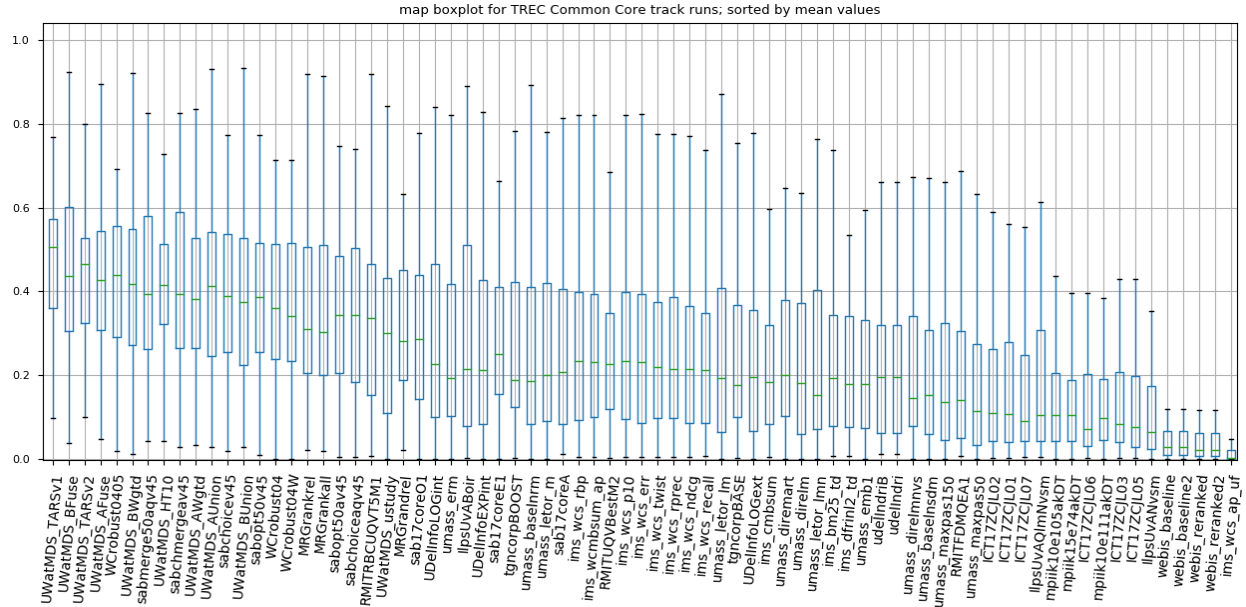


Figure 1: Box-plot of AP values over the 50 NIST queries.

6 Analysis

In what follows we perform a preliminary analysis regarding the submitted runs, the 50 NIST topics, and the relevance judgments obtained.

6.1 Uniquely identified relevant documents

Table 3 shows the number of uniquely retrieved relevant documents per team. Here, unique relevant document is defined using the following criteria: 1. consider only the top 100 ranks as the retrieved set; 2. a document retrieved by multiple runs of the same group is counted as unique as long as no other group retrieved it; 3. only judged runs are used in the computation; 4. total is over all topics and all judged runs of the group.

Table reftab:unique-run shows the number of uniquely retrieved relevant documents per run. Here, unique relevant document is defined using the following criteria: 1. consider only the top 100 ranks as the retrieved set; 2. a document retrieved by multiple runs of the same group is counted as unique as long as no other group retrieved it; 3. only judged runs are used in the computation; 4. total is over all topics of each single run of the group. This really means that the table counts how many unique relevant documents a team would have contributed if they had just submitted the one run under consideration. Table reftab:unique-run also categorizes runs on the basis of manual vs. automatic and use of TREC Robust track labels or not.

Last, the numbers in Table 5 are computed in the same way as in Table 4, however this time only automatic runs without any use of Robust track labels were considered, to illustrate a situation in which no prior judgments is available, while at the same time manual effort is completely avoided. sab17coreA and llpsUvANvsm contribute most of the unique relevant documents in this situation, with the latter run using a latent neural algorithm.

	automatic	manual	mixture	Total
BASELINE:	12	0	0	12
ICTNET:	39	0	0	39
MPIID5:	20	0	0	20
MRG_UWaterloo:	0	208	0	208
RMIT:	2	229	3	234
Sabir:	694	0	0	694
UMass:	24	0	0	24
UWaterlooMDS:	0	683	0	683
UvA_ILPS:	84	0	0	84
WaterlooCormack:	286	0	0	286
Webis:	17	0	0	17
ims-core:	8	0	0	8
tgncorp:	0	203	0	203
udel:	1	0	0	1
udel_fang:	36	0	0	36

Table 3: Totals per group over 50 NIST-judged topics of unique relevant retrieved.

Participating run	manual	labels	unique	Participating run	manual	labels	unique
umass_baselnrm			2	UWatMDS_BFuse	✓		161
umass_baselnsdm			2	UWatMDS_BUnion	✓		193
ims_bm25_td			4	UWatMDS_BWgtd	✓		209
ims_dfrinl2_td			8	UWatMDS_ustudy	✓		216
ICT17ZCJL01			9	UWatMDS_TARSv1	✓		329
ICT17ZCJL02			13	IlpsUvABoir		✓	19
ICT17ZCJL03			20	IlpsUvANvsm			66
ICT17ZCJL06			17	IlpsUvAQImNvsm			55
ICT17ZCJL07			16	WCrobust04		✓	170
mpiik10e105akDT		✓	18	WCrobust0405		✓	193
mpiik10e111akDT		✓	18	WCrobust04W		✓	165
mpiik15e74akDT		✓	17	webis_baseline			17
MRGrandrel	✓		168	webis_baseline2			17
MRGrankall	✓		103	webis_reranked			17
MRGrankrel	✓		104	webis_reranked2			17
RMITFDMQEA1			6	ims cmbsum			7
RMITRBCUQVT5M1	✓	✓	125	ims_wcmbsum_ap			7
RMITUQVBESTM2	✓	✓	145	ims_wcs_ap_uf			0
sab17coreA			48	ims_wcs_ndcg			6
sab17coreE1		✓	274	ims_wcs_p10			7
sab17coreO1		✓	279	tgncorpBASE	✓		150
sabchmergeav45		✓	92	tgncorpBOOST	✓		158
sabchoiceaqv45		✓	202	udelIndri			1
umass_direlm			0	udelIndriB			1
umass_direlmnvs			8	UDeInfoEXPint		✓	28
umass_diremart			7	UDeInfoLOGext		✓	10
umass_maxpas150			4	UDeInfoLOGint		✓	19
umass_maxpas50			12				

Table 4: Totals per run over 50 NIST-judged topics of unique relevant retrieved.

Automatic run w/o labels	unique	Automatic run w/o labels	unique
umass_baselnrm	14	udelIndri	22
umass_baselnsdm	7	udelIndriB	22
ims_bm25_td	39	umass_direlm	25
ims_dfrinl2_td	45	umass_direlmnvs	88
ICT17ZCJL01	53	umass_diremart	84
ICT17ZCJL02	65	umass_maxpas150	44
ICT17ZCJL03	94	umass_maxpas50	96
IlpsUvANvsm	223	ICT17ZCJL06	79
IlpsUvAQImNvsm	168	ICT17ZCJL07	74
ims_cmbsum	33	ims_wcs_ndcg	33
ims_wcmbsum_ap	34	ims_wcs_p10	37
ims_wcs_ap_uf	7	webis_baseline	43
RMITFDMQEA1	26	webis_baseline2	43
sab17coreA	240	webis_reranked	43
		webis_reranked2	43

Table 5: Total number of unique relevant documents per automatic run (making no use of past labels) over 50 NIST-judged topics.

6.2 Manual vs. Automatic with and without use of past labels

Figure 2 demonstrates the mean performance of the contributing to the pool runs measured by MAP, colored on the basis of their categorization, while Figure 3 summarizes the performance of the four categories of runs with a boxplot of MAP. As it can be observed in both figures, Manual runs and runs that make use of the Robust labels outperform on average the fully automatic runs, as expected. Interestingly enough, manual runs that make use of labeled data perform on average worse than manual runs that do not make use of labels, however no safe conclusions can be drawn here given that many other parameters and algorithms change across the different groups of runs.

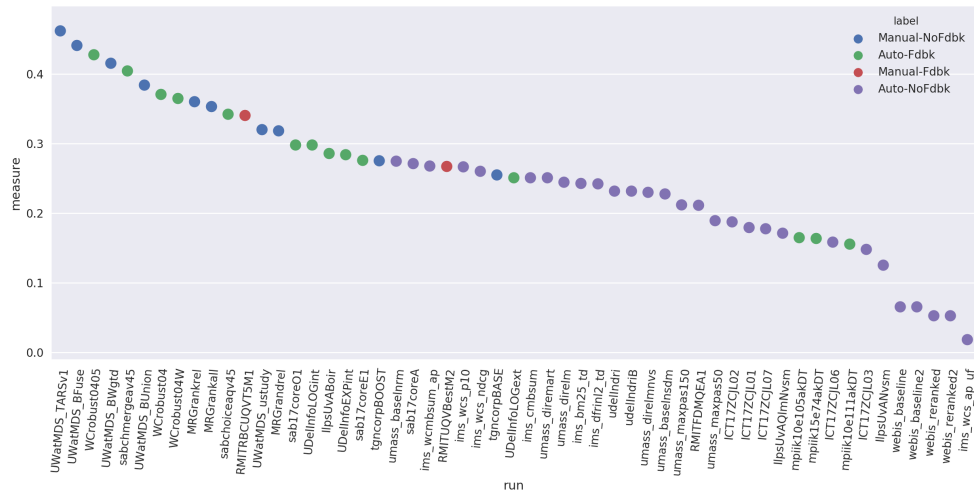


Figure 2: Scatter plot of MAP values over the 50 NIST queries. Each circle mark represents a run. The figure includes only the runs that contributed to the pool.

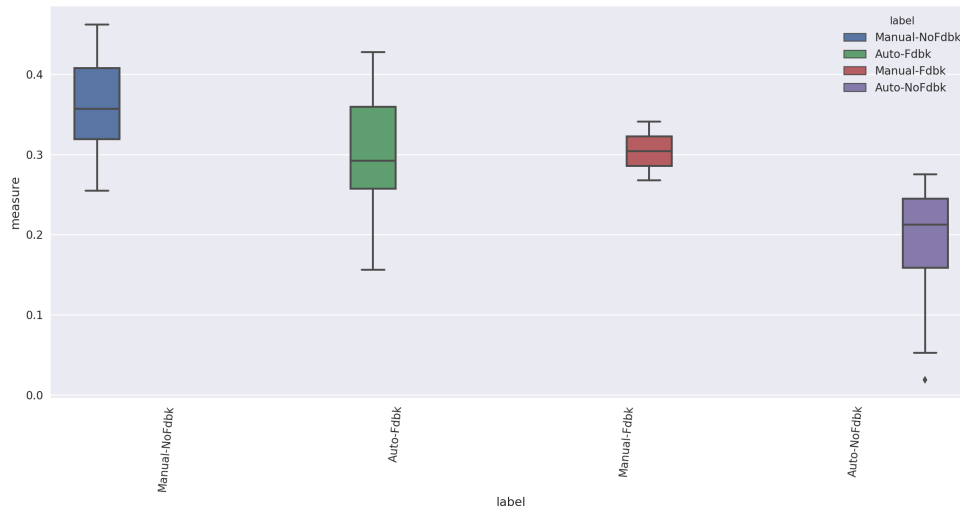


Figure 3: Box-plot of MAP values over the 50 NIST queries. Each box-plot represents the MAP values of a set of runs under the same category: (a) automatic, without making use of past labels (Auto-NoFdbk), (b) automatic, making use of past labels (Auto-Fdbk), (c) manual, without making use of past labels (Manual-NoFdbk), and (d) manual, making use of past labels (Manual-Fdbk). The figure includes only the runs that contributed to the pool.

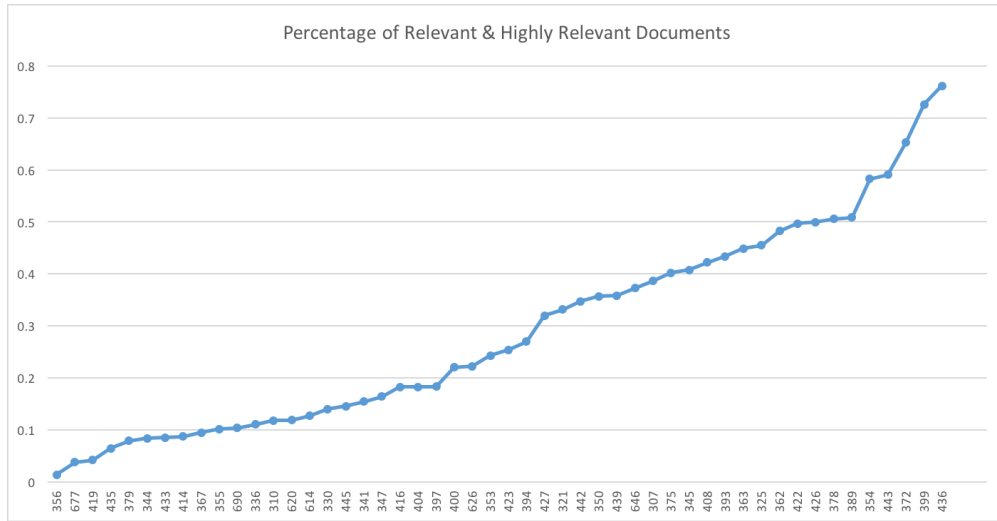


Figure 4: Percentage of relevant and highly relevant documents per topic.

6.3 Analysis of Topics

A total of 30030 query-document pairs were judged by NIST assessors, with 21028 of them found not relevant, 5549 found relevant, and 3453 highly relevant. On average 30% of the documents were found relevant or highly relevant. Figure 4 shows the percentage of relevant and highly relevant documents per topic.

7 Analysis of Judgments

Leave-one-out (LOO) experiments are one measure of collection quality. In a LOO experiment, the unique relevants for a target team are removed from the qrels and all runs are evaluated using this reduced qrels. The Kendall's τ correlation between the ranking of systems produced by the original qrels and the ranking of systems produced by the reduced qrels is then computed. Table 7 shows the results for the Common Core track when each participating team in turn is used as the target and MAP is the measure used to rank systems. The table also gives the number of unique relevant documents retrieved by the team. The final column in the table shows how individual runs were affected. For example, '-4 -3 -2 -2 -2 -1' means one run dropped four spots in the reduced qrels ranking, another dropped three spots, three runs dropped two spots, and a final run dropped one spot. Only the drops are listed (when one run drops at least one other must rise). The Kendall's τ scores are all well above 0.95 showing there is not much change in the rankings overall. Nonetheless, five runs dropped by seven or more spots, and one run by 18. For each of these five runs, the run was submitted by the team whose unique relevants were removed.

Team	Number Unique	Kendall's τ	Ranks Dropped
BASELINE	12	1.0	
ICTNET	39	1.0	
ims-core	8	0.999	-1
MPIID5	20	0.999	-1
MRG_UWaterloo	208	0.994	-2 -2 -1 -1 -1 -1 -1
RMIT	234	0.994	-8
Sabir	694	0.967	-18 -9 -4 -3 -2 -1 -1 -1 -1 -1 -1 -1 -1 -1
tgncorp	203	0.985	-11 -7 -2 -1
udel	1	1.0	
udel_fang	36	0.999	-2
UMass	24	0.999	-1
UvA.ILPS	84	1.0	
UWaterlooMDS	683	0.979	-3 -3 -3 -3 -3 -2 -2 -2 -2 -2 -2 -1 -1
WaterlooCormack	286	0.990	-4 -3 -2 -2 -2 -1
Webis	17	1.0	

Table 6: LOO Results for Common Core Track Runs. Given are the number of unique relevant documents retrieved by a team over all 50 topics and the Kendall's τ correlation between system rankings formed when evaluating all submissions by MAP using the original qrels and the qrels formed by removing the team's unique relevants. The last column shows how many ranks some run dropped when evaluated using the reduced qrels. The largest drops were runs submitted by the team whose unique relevants were removed.

8 Conclusions

The TREC 2017 Common Core track provided a traditional ad-hoc retrieval task to participants. The track attracted a diverse set of runs, including manual runs, but also state-of-the-art neural retrieval runs. A number of pooling methods were tested prior to the construction of the collection. Leave-one-out experiments yield a high value of Kendall's τ , nevertheless a small number of runs were penalized when they were left out of pooling. The manuscript will be updated with further analysis, and the results of the on-going crowdsourcing experiment.

References

- Chris Buckley, Darrin Dimmick, Ian Soboroff, and Ellen Voorhees. Bias and the limits of pooling for large collections. *Information Retrieval*, 10:491–508, 2007.
- David E. Losada, Javier Parapar, and Álvaro Barreiro. Feeling lucky? Multi-armed bandits for ordering judgements in pooling-based evaluation. In *Proceedings of SAC 2016*, pages 1027–1034, 2016.
- Emine Yilmaz, Evangelos Kanoulas, and Javed A. Aslam. A simple and efficient sampling method for estimating AP and NDCG. In *Proceedings of the Thirty-First Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2008)*, pages 603–610, 2008.