The TREC 2001 Filtering Track Report

Stephen Robertson
Microsoft Research
Cambridge, UK
ser@microsoft.com

Ian Soboroff
NIST
Gaithersburg MD, USA
ian.soboroff@nist.gov

Abstract

The TREC–10 filtering track measures the ability of systems to build persistent user profiles which successfully separate relevant and non-relevant documents. It consists of three major subtasks: adaptive filtering, batch filtering, and routing. In adaptive filtering, the system begins with only a topic statement and a small number of positive examples, and must learn a better profile from on-line feedback. Batch filtering and routing are more traditional machine learning tasks where the system begins with a large sample of evaluated training documents. This report describes the track, presents some evaluation results, and provides a general commentary on lessons learned from this year's track.

1 Introduction

A text filtering system sifts through a stream of incoming information to find documents relevant to a set of user needs represented by profiles. Unlike the traditional search query, user profiles are persistent, and tend to reflect a long term information need. With user feedback, the system can learn a better profile, and improve its performance over time. The TREC filtering track tries to simulate on-line time-critical text filtering applications, where the value of a document decays rapidly with time. This means that potentially relevant documents must be presented immediately to the user. There is no time to accumulate and rank a set of documents. Evaluation is based only on the quality of the retrieved set.

Filtering differs from search in that documents arrive sequentially over time. The TREC filtering track consists of three subtasks: adaptive filtering, batch filtering, and routing. In adaptive filtering, the system starts with only a user profile and a very small number of positive examples (relevant documents). It must begin filtering documents without any other prior information. Each retrieved document is immediately judged for relevance, and this information can be used by the system to adaptively update the filtering profile. In batch filtering and routing, the system starts with a large set of evaluated training documents which can be used to help construct the search profile. For batch filtering, the system must decide to accept or reject each document, while routing systems can return a ranked list of documents. The core tasks for TREC–10 are very similar to those investigated in TREC–7 through TREC–9.

Traditional adhoc retrieval and routing simulate a non-interactive process where users look at documents once at the end of system processing. This allows for ranking or clustering of the retrieved set. The filtering model is based on the assumption that users examine documents periodically over time. The actual frequency of user interaction is unknown and task-dependent. Rather than create a complex simulation which includes partial batching and ranking of the document set, we make the simplifying assumption that users want to be notified about interesting documents as soon as they arrive. Therefore, a decision must be made about each document without reference to future documents, and the retrieved set is ordered by time, not estimated likelihood of relevance. The history and development of the TREC Filtering Track can be traced by reading the yearly final reports:

- TREC–9 http://trec.nist.gov/pubs/trec9/t9_proceedings.html (#3) [8]
- TREC–8 http://trec.nist.gov/pubs/trec8/t8_proceedings.html (#3 - 2 files) [4]
- TREC–7 http://trec.nist.gov/pubs/trec7/t7_proceedings.html (#3 - 2 files) [3]
- TREC–6 http://trec.nist.gov/pubs/trec6/t6_proceedings.html (#4 and #5) [2]
- TREC–5 http://trec.nist.gov/pubs/trec5/t5_proceedings.html (#5) [7]
- TREC–4 http://trec.nist.gov/pubs/trec4/t4_proceedings.html (#11) [6]

Information on the participating groups and their filtering systems can be found in the individual site reports, also available from the TREC web site.

2 TREC–10 Task Description

For those familiar with previous TRECs, the basic filtering tasks in TREC–10 are similar to those investigated in TREC–7 through TREC–9. The batch-adaptive task has been abandoned (in the interests of reducing the number of different tasks in the track), and a new evaluation measure has been introduced, in lieu of the target-based measure introduced in TREC–9. The corpus and topics are again somewhat different from those used previously. In this section, we review the corpus, the three sub-tasks, the submission requirements, and the evaluation measures. For more background and motivation, please consult the TREC–7 track report [3].

2.1 Data

For the second year, the TREC–10 filtering experiments went outside the usual TREC collections. The new corpus recently provided by Reuters for research purposes [5] was used. This is a collection of about 800,000 news stories, covering a time period of a year in 1996-7. The items are categorised according to a standard set of Reuters categories, some of which were selected as discussed below to form the "topics" for filtering (in a similar fashion to the way MeSH headings were used in TREC–9).

Items in the collection have unique identifiers and are dated but not timed. For the purpose of the experiment, it is assumed that the time-order of items within one day is the same as identifier order. (Item id on its own is insufficient for ordering, as there is some conflict across days). The first 12 days' items, 20 through 31 August 1996, were taken as the training set (which could be used in ways specified below). The remainder of the collection formed the test set.

The category codes applied by Reuters are of three kinds: topic, region and industry. Only the topic codes were used; as with MeSH codes last year, the idea was to treat these categories as topics in the TREC sense, and regard the assignment of category codes by Reuters indexers as relevance judgements. One problem was the range of frequencies of assignment – some codes are extremely rare and some are applied to a substantial proportion of the collection. It was decided to limit this range at both ends, on the basis of the training set. Any code that occurred in more than 5% of the training set was rejected (this is too far removed from the usual TREC topic relevance set size). Also, since the tasks required some relevance data in the training set, specifically 2 relevant items for adaptive filtering, any code that occurred not at all or once only in the training set was rejected. The remaining 84 codes formed the basis for the filtering topics.

The Reuters Corpus is supplied with a list of codes, with a short text heading or description for each one (generally 1–3 words), and the numerical code. The numbering of the codes implies some hierarchical structure. Participants were provided with the text heading as the text of each topic (in the style of TREC 'short topics'), but were also able to make use of the hierarchical relations implied.

2.2 Tasks

The adaptive filtering task is designed to model the text filtering process from the moment of profile construction. In TREC–10, following the idea first used in TREC–9, we model the situation where the user arrives with a small number of known positive examples (relevant documents). For each topic, a random sample of two of the relevant documents in the training set was selected and made available to the participants for this purpose; no other relevance judgements from the training set could be used. Subsequently, once a document is retrieved, the relevance assessment (when one exists) is immediately made available to the system. Unfortunately, it is not feasible in practice to have interactive human assessment by NIST. Instead, assessment is simulated by releasing the pre-existing relevance judgement for that document. Judgements for unretrieved documents are never revealed to the system. Once the system makes a decision about whether or not to retrieve a document, that decision is final. No back-tracking or temporary caching of documents is allowed. While not always realistic, this condition reduces the complexity of the task and makes it easier to compare performance between different systems.

Systems are allowed to use the whole of the training set of documents (but no other relevance judgements than the two provided for each topic) to generate collection frequency statistics (such as IDF) or auxiliary data structures (such as automatically-generated thesauri). Resources outside the Reuters collection could also be used. As documents were processed, the text could be used to update term frequency statistics and auxiliary document structures even if the document was not matched to any profile. Groups had the option to treat unevaluated documents as not relevant.

In batch filtering, all the training set documents and all relevance judgements on that set are available in advance. Once the system is trained, the test set is processed in its entirety (there was no batch-adaptive task in TREC–10). For each topic, the system returns a single retrieved set. For routing, the training data is the same as for batch filtering, but in this case systems return a ranked list of the top 1000 retrieved documents from the test set. Batch filtering and routing are included to open participation to as many different groups as possible.

2.3 Evaluation and optimisation

For the TREC experiments, filtering systems are expected to make a binary decision to accept or reject a document for each profile. Therefore, the retrieved set consists of an unranked list of documents. This fact has implications for evaluation, in that it demands a measure of effectiveness which can be applied to such an unranked set. Many of the standard measures used in the evaluation of ranked retrieval (such as average precision) are not applicable. Furthermore, the choice of primary measure of performance will impact the systems in a way that does not happen in ranked retrieval. While good ranking algorithms seem to be relatively independent of the evaluation measure used, good classification algorithms need to relate very strongly to the measure it is desired to optimise.

Two measures were used in TREC–10 for this purpose (as alternative sub-tasks). One was essentially the linear utility measure used in previous TRECs, and described below. The other was new to the track for TREC–10: it is a version of the van Rijsbergen measure of retrieval performance.

F-beta

This measure, based on one defined by van Rijsbergen, is a function of recall and precision, together with a free parameter beta which determines the relative weighting of recall and precision. For any beta, the measure lies in the range zero (bad) to 1 (good). For TREC 2001, a value of beta=0.5 has been chosen, corresponding to an emphasis on precision (beta=1 is neutral). The measure (with this choice of beta) may be expressed as follows:

$$\text{T10F} = \frac{1.25 \times \text{No. of relevant retrieved docs}}{\text{No. of retrieved docs} + 0.25 \times \text{No. of relevant docs}}$$

(T10F is set to zero if the number of retrieved documents is zero.)

Linear utility

The idea of a linear utility measure has been described in previous TREC reports (e.g. [4]). The particular parameters being used are a credit of 2 for a relevant document retrieved and a debit of 1 for a non-relevant document retrieved:

$$\text{T10U} = 2R^{+} - N^{+}$$

which corresponds to the retrieval rule:

$$\text{retrieve if } P(\text{rel}) > .33$$

Filtering according to a utility function is equivalent to filtering by estimated probability of relevance; the corresponding probability threshold is shown.

When evaluation is based on utility, it is difficult to compare performance across topics. Simple averaging of the utility measure gives each retrieved document equal weight, which means that the average scores will be dominated by the topics with large retrieved sets (as in micro-averaging). Furthermore, the utility scale is effectively unbounded below but bounded above; a single very poor query might completely swamp any number of good queries.

For the purpose of averaging across topics, the method used for TREC-10 is as in TREC–8. That is, the topic utilities are scaled between limits, and the scaled values are averaged. The upper limit is the maximum utility for that topic, namely

$$\text{MaxU} = 2 \times (\text{Total relevant})$$

The lower limit is some negative utility, MinU, which may be thought of as the maximum number of non-relevant documents that a user would tolerate, with no relevants, over the lifetime of the profile. If the T10U value falls below this minimum, it will be assumed that the user stops looking at documents, and therefore the minimum is used.

$$\text{T10SU} = \frac{\max(\text{T10U}, \text{MinU}) - \text{MinU}}{\text{MaxU} - \text{MinU}}$$

for each topic, and

$$\text{Mean T10SU} = \text{Mean T10SU over topics}$$

Different values of MinU may be chosen: a value of zero means taking all negative utilities as zero and then normalising by the maximum. A value of minus infinity is equivalent (in terms of comparing sytems) to using unnormalised utility. The primary evaluation measure for TREC–10 has

$$\text{MinU} = -100$$

Other measures

In the official results tables, a number of measures are included as well as the measure for which any particular run was specifically optimised. The range is as follows:

For adaptive and batch filtering:

- Macro average recall
- Macro average precision
- Mean T10SU (scaled utility) over topics, over the whole period and broken down by time period for adaptive filtering
- Mean T10F (F-beta) over topics. Note that this is referred to as F-beta, but has beta set to 0.5, as above
- Zeros, that is, the number of topics for which no documents were retrieved over the period.

For routing: the usual range of ranked-output performance measures, and the number of topics which scored P@1000 > 0.9.

2.4 Submission Requirements

Each participating group could submit a limited number of runs, in each category: Adaptive filtering 4; Batch filtering 2; Routing 2.

Any of the filtering runs could be optimised for either T10F or T10SU – a declaration was required of the measure for which each run was optimised. There were no required runs, but participants were encouraged to provide an adaptive filtering run with T10SU optimisation.

Groups were also asked to indicate whether they used other parts of the TREC collection, or other external sources, to build term collection statistics or other resources. It was also possible to make limited use of other Reuters data – again, groups were asked to declare this.

3 TREC–10 results

Nineteen groups participated in the TREC–10 filtering track (five more than in TREC–9) and submitted a total of 66 runs (slightly less than last time, because of the substantially reduced number of options). These break down as follows: 12 groups submitted adaptive filtering runs, 10 submitted to batch filtering, and 11 to routing.

Here is a list of the participating groups, including abbreviations and run identifiers. Participants will generally be referred to by their abbreviations in this paper. The run identifiers can be used to recognize which runs belong to which groups in the plotted results.

	Abbreviation	Run identifier
Johns Hopkins University Applied Physics Lab	apl-jhu	apl10
Fudan University	Fudan	FDUT10
IRIT-SIG	IRIT	mer10
Justsystem Corporation	Justsystem	jscbtafr
Korea Advanced Institute of Science and Technology	KAIST	KAIST10
David D. Lewis, Independent Consultant	Lewis	DLewis01
Moscow Medical Academy	MCNIT	MMAT10
SER Technology Deutschland GmbH	SER	ser
Tampere University of Technology	Tampere	Visa
Chinese Academy of Sciences	chinese_academy	ICTAdaFT10
Clairvoyance Corporation	clairvoyance	CL01
Carnegie Mellon University	cmu-cat	CMUCAT
Carnegie Mellon University	cmu-lti	CMUDIR
Kent Ridge Digital Labs	kent_ridge	krdlT10
Microsoft Research Ltd	microsoft	ok10
Katholieke Universiteit Nijmegen	KUN	KUN
Oracle	oracle	ora
Rutgers University	rutgers-kantor	RU
University of Iowa	uiowa	UIowa

3.1 Summary of approaches

These brief summaries are intended only to point readers towards other work. Not all groups have a paper in the proceedings.

JHU APL participated in the routing and batch filtering tasks. Their routing runs used statistical language modeling with either character n-gram, stem, or word features. For batch filtering, they used support vector machines with different choices of feature vectors, kernels, score thresholding, and training skew factors.

Fudan University participated in the batch and adaptive filtering tasks. Their filtering profiles were a Rocchio-style weighted sum of positive training documents, with mutual information used to select terms. For adaptive filtering, their filtering procedure added pseudo-relevant documents to the initial profile, and modified both the profile and the threshold using positive and negative feedback.

IRIT-SIG participated in the routing, batch, and adaptive filtering tasks. Their Mercure system uses a spreading activation network to compute a relevance score. For routing and batch filtering, they trained their profiles using backpropagation. For adaptive filtering, they began with simple term-frequency weighted profiles and adapted both the profile and the decision threshold.

KAIST participated in the batch filtering task. They clustered the training documents in each topic into subtopics, trained a support vector machine for each subtopic, and OR'ed the binary classifier outputs to form a final decision for a topic.

David Lewis participated in the routing and batch filtering tasks. He used support vector machines with different weights for positive and negative training examples. The weighting parameter was chosen using n-fold cross-validation.

SER Technology participated in all three tasks. They used their commercial text classification system. For adaptive filtering they kept a constant decision threshold, and retrained their classifier using the top documents seen so far.

Tampere University of Technology participated in the routing task. They employed a novel profile learning technique that encodes feature terms and bins the coded values according to a statistical distribution, yielding a histogram for each word and sentence in a training document.

Chinese Academy of Sciences (CAS-ICT) participated in the adaptive filtering task. They used a vector-space approach with a profile adaptation function similar to Rocchio.

Clairvoyance participated in all three tasks. For batch filtering, they experimented with "ensembles" of simple filters in parallel or series rather than a single monolithic profile. For adaptive filtering, they used the same system as in TREC-8 to explore how well it performed on the new Reuters data.

The CMUCAT group from Carnegie Mellon University participated in the batch and adaptive filtering tasks. They compared kNN classification to the standard Rocchio method, and further explored issues with the utility metric.

The CMUDIR group from CMU participated in the adaptive filtering task. The used a language modeling approach to learn the maximum likelihood of the relevant documents discovered during filtering.

Microsoft Research Cambridge participated in the adaptive filtering task. They used their Keenbow system from last year, adding optimisation for the F-beta measure.

Katholieke Universiteit Nijmegen participated in all three tasks. Their system uses a version of Rocchio which decays terms in the profile over time, and a threshold optimisation method based on EM that models the distributions of the scores of relevant and non-relevant documents.

Oracle participated in all three tasks. They used the Oracle Text RDBMS/text retrieval system. They assigned concepts from a thesaurus to documents, summed the concept vectors from relevant training documents and selected the best concepts to represent each topic.

3.2 Evaluation results

Some results for the various participating groups are presented in the following graphs. Figures 1 and 2 show the adaptive filtering results for the two optimisation measures. In each graph, the horizontal line inside a run's box is the median topic score, the box shows interquartile distance, the whiskers extend to the furthest topic within 1.5 times the interquartile distance, and the circles are outliers.

Figure 3 shows the adaptive filtering utility scores broken down into four document periods, to illustrate learning effect. For readability, only the run with the best overall mean T10SU is shown. The official results pages for the adaptive task show this data for every run.

Figures 4 and 5 show the utility and F-beta results for batch filtering. Figure 6 shows mean uninterpolated average precision for routing. Note that while there is still a range of performance, in general scores were quite high in batch filtering and routing.

3.3 Utility Scaling

In their TREC-9 paper, Ault and Yang proposed that the track use the F-beta measure instead of T9P, but did not suggest a replacement for the T9U utility measure [1]. This year, they did propose such a measure, called *normalized filtering utility*, which following our notation and filtering costs, is:

$$U_f = \frac{2R^+ - N^+}{\text{MaxU}}$$

$$U'_f = \frac{\max(U_f, U_{f,min}) - U_{f,min}}{1 - U_{f,min}} \quad (U_{f,min} = -0.5)$$

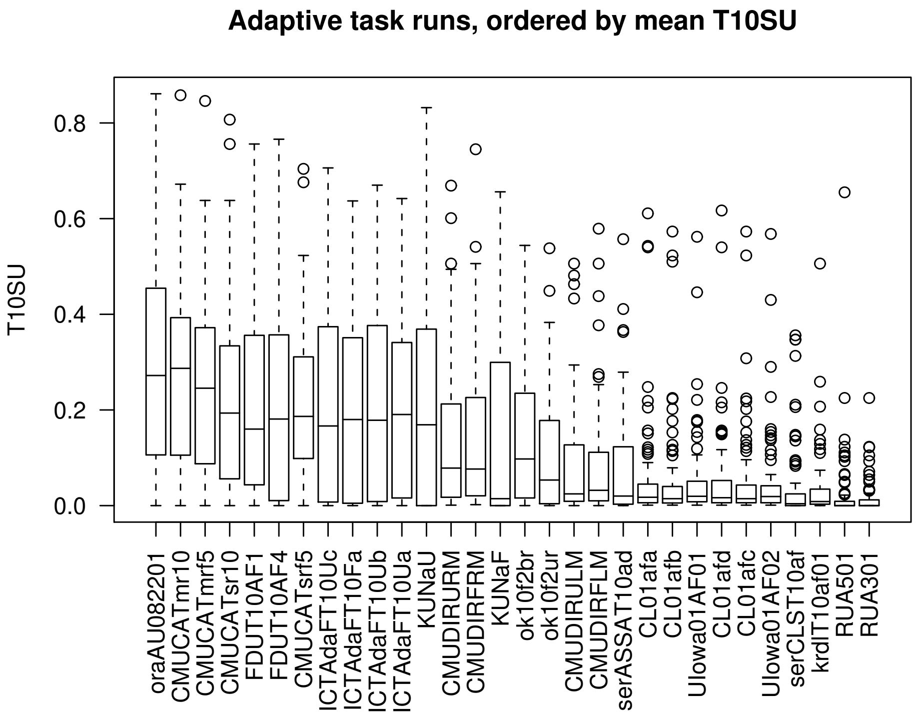

Figure 1: Adaptive filtering – T10SU

Note that this is not exactly Ault and Yang's notation; please see their paper elsewhere in this volume for details. The key difference is in how the measure is scaled between best possible and worst acceptable utility. To further discussion on utility metrics, we show the normalized filtering utility scores for adaptive filtering in Figure 7.

4 General Commentary

One major impression from the results of the TREC–10 filtering track is that the data set is significantly different in the way it behaves from those used in previous years. And the major single *a priori* difference, obvious from looking at the data set, is the number of relevant documents for each topic. There are topics which have in total tens of thousands of relevant documents, and most have hundreds or thousands. This is clearly a function of the way the topics were constructed, from Reuters categories, which tend to be fairly general (compared to typical/conventional TREC topics). Note that this is not simply a result of using predefined categories: in TREC–9, we used MeSH headings in a Medline document collection in roughly the same way. The topics defined from MeSH headings did not then appear to be very different from the more conventionally constructed ones. But MeSH headings are typically very much more specific than Reuters categories.

One effect of the use of these broad categories seems to have been as follows. In previous years, it has been very important for good performance to strictly limit the number of documents retrieved – it was all too easy to get into a non-recoverable negative utility realm by retrieving too many documents at the beginning. It is likely that some participants' thresholding methods were simply too restrictive for the conditions of the present test collection.

Nevertheless, other participants in adaptive filtering turned in very good performances. Even

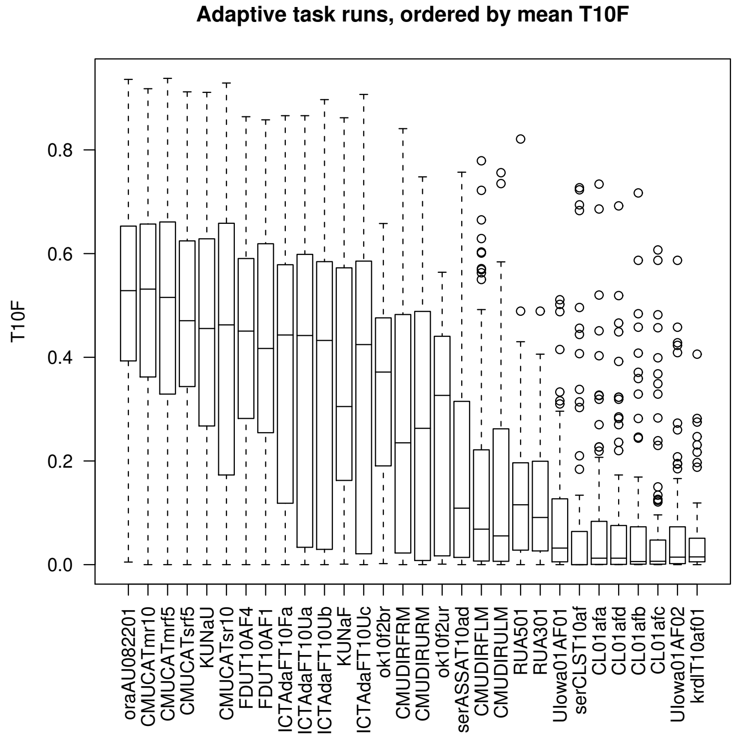

Figure 2: Adaptive filtering – T10F

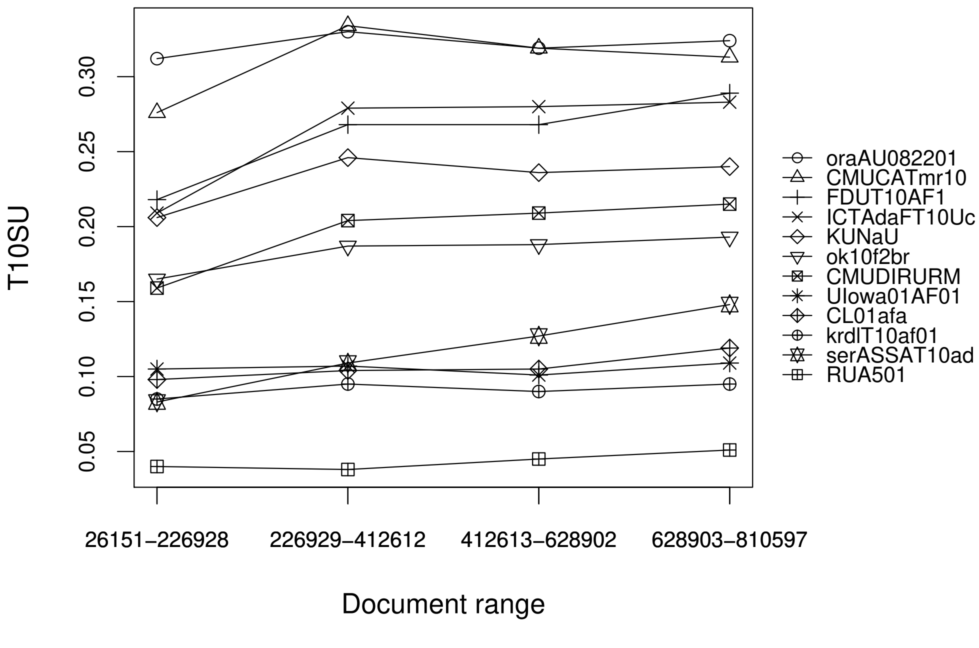

Figure 3: Adaptive filtering – T10SU within document period. The x axis labels show the document ID range in each period.

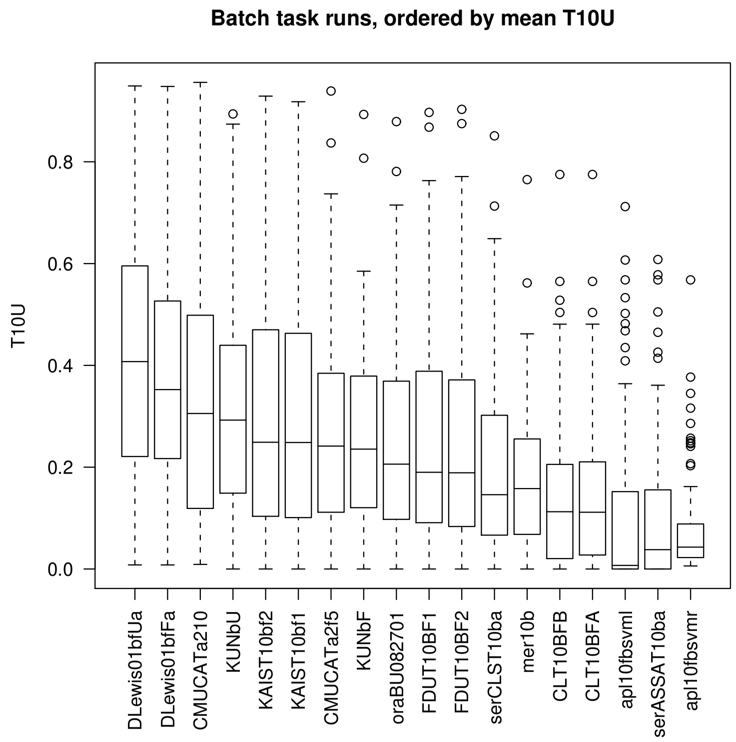

Figure 4: Batch filtering – T10SU

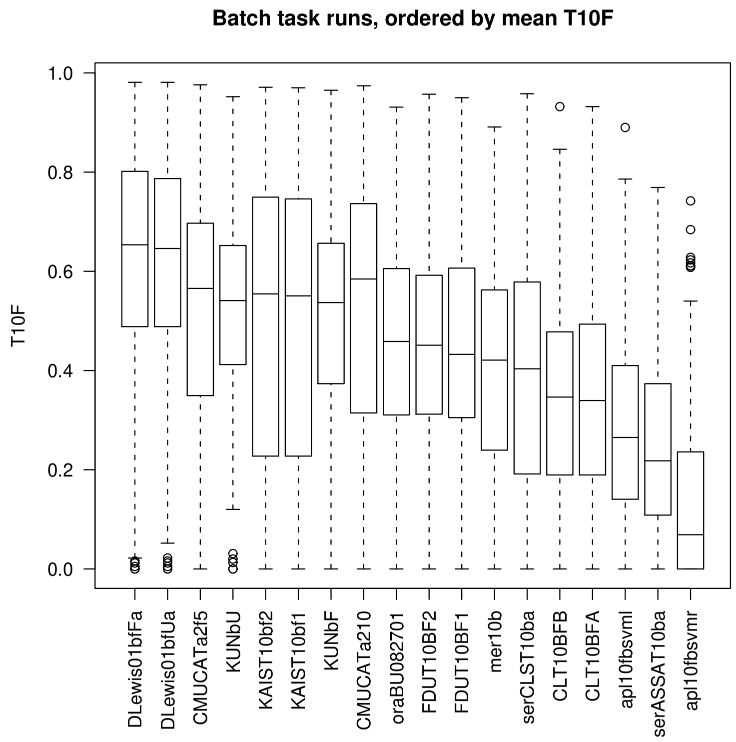

Figure 5: Batch filtering – T10F

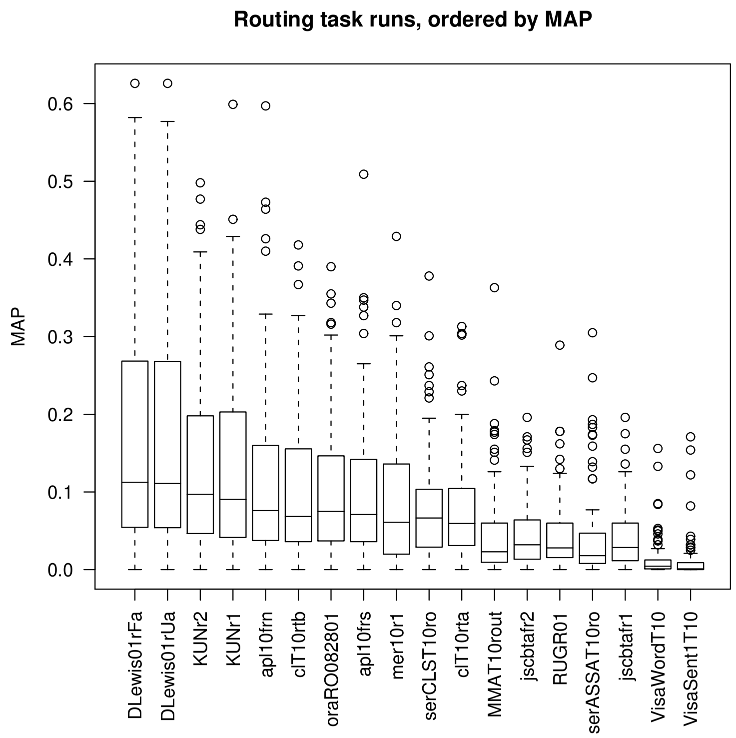

Figure 6: Routing – Mean Average Precision

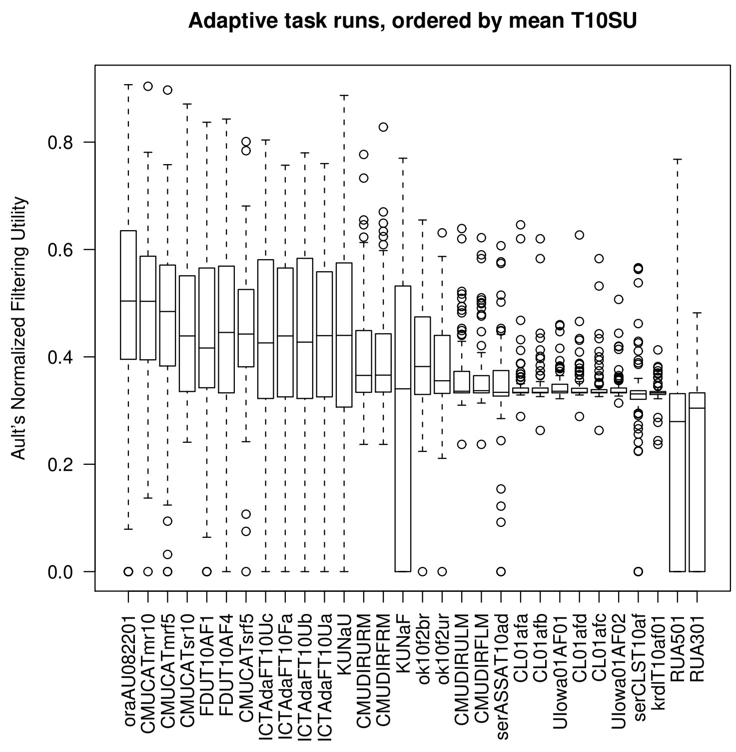

Figure 7: Adaptive filtering – Normalized Filtering Utility. The runs are ordered by mean T10SU, for comparison with Figure 1.

given the very limited starting point of 2 positive examples, it seems that a great deal of effective learning took place in the early part of the test period. The relatively flat learning curve over the whole learning period suggests not a lack of learning, but saturation. The best batch filtering and routing results also appear very good. A number of systems on a number of topics achieved over 90% precision at 1000 documents (this was the subject of some bets before TREC!).

This characteristic of the data set as defined for TREC–10 is seen as a disadvantage in a number of respects. One has to do with realism: while it may be possible to envisage situations in which a topic with 10,000 relevant documents over a year is plausible, it is well outside the sort of context we have typically imagined for a filtering system. Another is that it renders some measurements more questionable: if we can easily achieve 90% precision at 1000 documents, then that suggests that we should at a minimum evaluate much further down the ranking. This, however, would introduce logistical problems.

The likelihood is that the TREC–11 filtering track will use the Reuters corpus, but with a different set of topics. One aim would be to get back into a more reasonable range of numbers of relevant documents per topic.

Acknowledgements We give our thanks to all the people who have contributed to the development of the TREC filtering track over the years, in particular David Lewis, David Hull, Karen Sparck Jones, Chris Buckley, Paul Kantor, Ellen Voorhees, the TREC program committee, and the team at NIST.

References

[1] Ault, Tom and Yang, Yiming. kNN at TREC-9. In *The 9th Text Retrieval Conference (TREC-9), NIST SP 500-249*, pages 127–134, 2001.

[2] Hull, David A. The TREC-6 Filtering Track: Description and Analysis. In *The 6th Text Retrieval Conference (TREC-6), NIST SP 500-240*, pages 45–68, 1998.

[3] Hull, David A. The TREC-7 Filtering Track: Description and Analysis. In *The 7th Text Retrieval Conference (TREC-7), NIST SP 500-242*, pages 33–56, 1999.

[4] Hull, David A. and Robertson, Stephen. The TREC-8 Filtering Track final report. In *The 8th Text Retrieval Conference (TREC-8), NIST SP 500-246*, pages 35–56, 2000.

[5] The Reuters Corpus Volume 1 – see http://about.reuters.com/researchandstandards/corpus/

[6] Lewis, David. The TREC-4 Filtering Track. In *The 4th Text Retrieval Conference (TREC-4), NIST SP 500-236*, pages 165–180, 1996.

[7] Lewis, David. The TREC-5 Filtering Track. In *The 5th Text Retrieval Conference (TREC-5), NIST SP 500-238*, pages 75–96, 1997.

[8] Robertson, Stephen and Hull, David A. The TREC-9 Filtering Track final report. In *The 9th Text Retrieval Conference (TREC-9), NIST SP 500-249*, pages 25–40, 2001.