# Report on the TREC-5 Confusion Track

Paul B. Kantor SCILS, Rutgers Ellen Voorhees NIST

**Abstract**

For TREC-5, retrieval from corrupted data was studied through retrieval of single target documents from a corpus which was corrupted by producing page images, corrupting the bit maps, and applying OCR techniques to the results. In general, methods which attempted a probabilistic estimation of the original clean text fare better than methods which simply accept corrupted versions of the query text.

## 1 History

The confusion track originated at an informal meeting held during TREC-3, stimulated by interest in the potential of various schemes for retrieval based on imperfect OCR applied to scanned legacy texts. The guiding idea was that even an imperfect translation of the image into text might support effective retrieval, especially if the retrieval were based on text representations not dependent upon the identification of terms. Participants in this first meeting included Mark Damashek (NSA), David Grossman (GMU), Fritz Nordby (then at Paracel), and Paul Kantor, who was selected, on the"grey hair" principle, to serve as spokesman.

At that first meeting a range of methods for approaching the problem were considered, including use of overlapping n-grams of varying length, and efforts to reverse engineer the transition probabilities of the corruption operator. It was agreed that the process should involve a corruption algorithm whose transition matrix was known only to the TREC organizers, and not to the participants. The tenor of this discussion led to the informal name "corruption track", which was subsequently Bowdlerized to the more presentable "Confusion Track".

For TREC-4 the track was managed in a very low-key fashion (by PBK), and lost one of its originators when Nordby moved to another position. The participants, whose results will not be summarized here, obtained results which were not judged to be exciting.

## 2 The TREC-5 Task

For TREC-5 the confusion track used a particular type of retrieval problem called *known-item searching*. A known-item search is a retrieval task that simulates a user seeking a particular, partially-remembered document in the collection. In contrast to a more standard retrieval search where the goal is to retrieve/rank the entire set of documents that pertain to a particular subject of interest, the goal in the known-item search is to retrieve one particular document.

Known-item searching is well-suited to the task of retrieving corrupted data. When document content is corrupted, low-frequency words such as proper nouns and technical terms are the most affected. Yet low-frequency words are high-content-bearing words, and are precisely the words likely to be used to locate a specific document. Thus known-item searches exercise the parts of the retrieval methodologies that the track is most interested in. As a bonus, the searches do not

- Use of solar power by the Florida energy office.
- Excessive mark up of zero coupon treasury bonds.
- I am looking for a document about the dismissal of a lawsuit involving Adventist Health Systems.
- I am looking for theft data on the Chevrolet Corsica.
- efforts to establish cooperative breeding programs for the yellow crowned amazon parrot.
- morphological similarities between different populations of saltwater crocodiles.

Figure 1: Example known-item topics from the TREC-5 confusion track

require relevance assessments. Clearly, this obviates the need for relevance assessor's time — a critical resource at NIST. But it also means the track can run with fewer participants: since TREC uses pooled results to approximate exhaustive relevance assessments, the quality of the relevance assessments depends on the diversity of the pool and hence on the number of participants.

Participants in the track were asked to rank the top 1000 documents per topic on each of three different versions of the *1994 Federal Register*: the correct copy, a scanned copy that had approximately a 5% character error rate, and a scanned copy that had approximately a 20% character error rate. The 20% error rate version was created by performing OCR on an image that had been downsampled from the original image.

Figure 1 shows some examples of the known-item topics used in the track. The topics were created by five NIST staff members who developed ten topics each. Different authors used different techniques to construct their topics (using an index of the collection to find unique words, starting with "interesting" documents and adding conditions to the topic to ensure uniqueness, etc.), but the authors did *not* pick specific words that they thought would be difficult for the OCR process. The authors made every effort to ensure that only one document was a legitimate answer to the topic, and there have been no reported problems with this assessment.

## 3 Overview of Retrieval Schemes

Five broadly different retrieval methods were applied in TREC-5. They form a progression in terms of the detail with which they attempt to discern the correct text underlying the corrupted version. The reader is urged to consult the respective papers by the individual participants found elsewhere in this proceedings for more details about the approaches.

**Rutgers SCILS APLab:** The Rutgers APLab group used Unix utilities to search for any string, in the corrupted text, matching any n-gram defined by stopping at word boundaries and using a sliding window of width 5 within longer words. After removal of stop words (using the SMART list) match was defined using a wild card character in regular expressions, which matches any set of 0,1, or 2 characters.

Each term in the query was expanded into all the "expanded" patterns that meet these rules. Thus "cat" is searched as all of "cat, ..at, c..t, ca..". The corrupted text was processed line by line. Every match contributed one point to the score of the current line. The final score for a text was the average of the line scores. Note that this scheme gives added weight to uncorrupted texts. Thus "cat" in the text would match all four of the patterns.

**Australian National University (ANU):** In this approach queries were "corrupted", based on corruption errors discovered in a small sample of clean and 5% degraded texts. In effect, this expanded the query by the addition of likely corrupt terms. The span scoring method was then applied to corrupted texts and expanded queries to produce the final results.

**George Mason University (GMU):** The GMU approach used one-step retrieval, based on overlapping 4-grams, including term boundaries. Queries were expanded by the addition of the most frequent n-grams occurring in the top 10 documents, and the top 20 n-grams were added to the query with weight 0.4. Document-query similarity scores were computed using a cosine measure, with an inverse document frequency metric.

**CLARITECH Corporation (CLARIT):** In the CLARITECH approach, stochastic methods were applied to the documents to correct corrupted words on a sentence-by-sentence basis. Federal register data from 1988 and 1989 was used to estimate word frequencies, and word-word transition (bi-gram) probabilities. Correction was applied *only* to words that did not match the lexicon (call these *c-words*). For each such c-word, up to 200 candidates were listed and ranked in order of "probability to match the corrupt word". The top 10 candidates to each c-word were retained for sentence processing. This processing seeks to minimize the total stress, over the sentence, of the transitions between consecutive words. Thus CLARIT produced a single sentence-based-maximum likelihood assignment for every c-word, as a basis for further retrieval. Further retrieval was accomplished using the standard CLARIT indexing and retrieval.

**Swiss Federal Institute of Technology (ETH):** The ETH group made use of several devices. Document score was computed using a pivot method (which controls the effect of overly long documents), and feature contributions which include the product of $\ln(f(\phi, q), d(\phi), (1 + \ln(f^*(\phi, d))$. Here $f(\phi, d)$ is the observed frequency of a feature $\phi$ in the document $d$, (queries $q$ are treated as documents also) and $f^*$ is a corrected estimated frequency. The latter is determined in three steps. First, the document is divided into overlapping slots, which might contain the feature. A slot is used for further computation if it contains at least a fraction P of the feature's characters. The number of slots such that the edit distance between the slot contents and the feature is less than 20% of the length of the feature is determined. The probability that a feature appears in a slot is set to a nominal value for each matching slot, and then summed over all the slots in the document. The sum is then multiplied by a constant which makes the estimate of feature frequency more accurate, as determined by regression analysis applied to a set of 100 documents used in clear, 5% and 20% corrupted forms. The features used are (Porter) stems, including the preceding white space.

In practice, this detailed calculation was carried out only for a set of 2000 documents for each query, retrieved by straight n-gram screening, using n=4 for 5% and 3 for 20% corrupted data.

In sum, the ETH method gives a document credit for all of the features which have a "sufficiently close match" to the noisy text as it is presented. All features are words or initial substrings of words, and the method could have trouble with corruption of the word separation characters.

As these brief descriptions show, the methods vary in their treatment of the query, and of the corrupted texts. They appear to form a progression in the following sense:

Rutgers expanded the terms appearing in the query by a 5-gram sliding window with each character replaced with any set of 0, 1 or 2 characters. The 5-grams did not cross word boundaries. Retrieval ranking was based on the average number of hits per line of text. This probably discriminates much too strongly against long documents. Performance was poor.

ANU expanded queries based on corruption errors found likely in a study of a sample of corrupted text. Thus additional terms (which might in principle be words in a lexicon) were added to the query.

GMU resolved both query and documents into overlapping 4-grams, judged to be more resistant to corruption, and required an exact match. Special stop-lists of 4-grams were constructed. Queries were expanded by a method based on preliminary retrieval from the corpus, resulting in the addition of new 4-grams.

These three methods represent expansion of the query, in an effort to include or match corrupted forms that either might (Rutgers), or could (ANU), or sometimes do (GMU) happen under the corruption observed. Results for the first of these methods were relatively weak; the second method was not applied to the most severely corrupted data, and the third exhibited somewhat surprising performance detailed below.

The remaining two methods sought to "expand" or "clarify" the corrupted texts.

CLARIT used statistical methods to replace each non-word by a word which makes the entire resulting sentence most likely in some well-defined sense. Each non-word is replaced by exactly one word.

ETH, in effect, replaced each "slot" (which might be occupied by a word in the corrupted text) by a vector of candidate words, each of which is permitted to contribute to the computed similarity to the question. This is, in principle, a wider expansion of the corrupted text, since the second ranked candidate can enter the computation in this method, but not in the CLARIT method.

# 4 Retrieval Results

Participants were asked to submit a ranking of the top 1000 documents for each topic. The runs are evaluated based on the rank given to the target document; no "partial credit" was given for retrieving documents similar to the target.

## 4.1 Evaluation Measures

Several different evaluation measures are given for each system in the confusion results in Appendix A. The first measure is the Raw Ranks table. This table gives the rank at which the known item was retrieved for each of the three versions of documents for all 49 topics[1]. A document that was not retrieved at all in the top 1000 documents was assigned a rank of 2000.

The "mean rank when found" and the "mean reciprocal rank" are given in the final rows of the Raw Ranks table. The mean rank when found is the mean rank at which the known item was found averaged across all topics that retrieved the known item in the top 1000 documents. This measure gives an easily-interpreted idea of how well the retrieval methodology ranks the known item if it finds it at all. (When the average is computed over all topics, this measure is also known as *expected run length*.)

[1] Topic 29 had to be dropped from the evaluation. Some input files got truncated when producing the degraded versions of the text, so all three collections were restricted to the smallest of the three sets. One of the omitted documents was the target item for topic 29.

The mean reciprocal rank is the mean of the reciprocal of the rank at which the known item was found over all the topics, using 0 (not 1/2000) as the reciprocal for topics that did not retrieve the known document. Unlike the mean rank when found measure, this measure penalizes runs that did not retrieve a known item while minimizing the difference between, say, retrieving a known item at rank 750 and retrieving it at rank 900. It is also bounded between 1 and 0, inclusive, so the measure is interpretable without knowing how many documents were ranked. Indeed, since there is only one relevant document per query, the reciprocal rank of that document is the precision at that document, and therefore it is the average precision of the query as well (average precision is the precision averaged over all relevant documents of the query). Average precision is a frequently used measure in the other parts of TREC, so "mean reciprocal rank" gives some basis of comparison with other retrieval methods.

A histogram of the ranks at which the known item was found is given in the second table in the appendix. It gives the number of topics for which the item was not retrieved at all, was found in the first ten ranks, was found in rank 11–100, and was retrieved but ranked greater than 100.

A graphical representation of the results is given below the histogram. The graph plots the cumulative percentage of queries whose known item has been found at each rank. For example, if the value of the curve is 27 at rank 15, then 27% of the topics had their known item found at rank 15 or lower.

### 4.2 Comparative Performance

As mentioned above, expected search length is not a good measure to use to compare systems because the largest contributions come from the nominal positions assigned to those documents which were not received. On the other hand, its reciprocal, which increases for better systems, does not have this problem. A Generalized Retrieval Operating Characteristic [1] can be used to obtain a more detailed view of comparative performance.

If we consider the cumulated value delivered by a set of ranked lists as proportional (with constant $v$) to the number ($G$) of target documents found (out of a total of $S$ target documents), and cost as proportional (with constant $c$) to the number of documents which must be examined before reaching them, the corresponding measure of value is:

$$\begin{aligned} V &= vG - c\textstyle\sum_{i\,found} r(i) - 1000(S-G)c \\ &= G(v+1000c) - c\textstyle\sum_{i\,found} r(i) - 1000S \end{aligned}$$

Hence systems would be ranked according to:

$$G(v+1000c) - c\textstyle\sum_{i\,found} r(i)$$

This is the same as ranking them according to:

$$G(\tfrac{v}{c}+1000) - \textstyle\sum_{i\,found} r(i)$$

In other words, the relative importance of finding a document at all, compared to the importance of placing found documents high in the list, depends in an unavoidable way on the cost assigned to examining documents. Thus there is no single measure which covers all reasonable opinions about this parameter.

On the other hand, one may extend the idea of Generalized Retrieval Operating Characteristic to compare systems. The extension is to imagine that all retrieved lists are perused in parallel, and to ask how many of the documents have been found, when $r$ documents have been examined on each list. This performance curve may be calculated directly from the reported lists as follows. Let $r(j)$ be the rank of the sought item, for topic $j$. We can compute the cumulated number of documents examined up to each "hit". In effect, we imagine that for a set of $S$ topics, there are $S$ analysts who work in parallel. Each examines the first document in the list for her problem. As soon as she finds the desired document, she stops working on this task. If she reaches the end of the list she stops working. The process continues until all the analysts have stopped working. We plot the number of good documents that have been discovered against the number of documents that have been examined in all rounds prior to its discovery.

If the curve for one scheme lies everywhere above the curve for another then, at least for this set of target documents, it delivers greater value, whatever the value ($v$) assigned to good and cost ($c$) assigned to bad documents. If neither curve is always above the other, then we cannot definitely state that one scheme is to be preferred to another. As with other measures currently used in information retrieval evaluation, the statistical significance (confidence levels, confidence intervals, etc.) this type of comparison is not known, particularly for the case of multiple comparisons.

The overall confusion track results roughly parallel the order of generality of the nets cast by the methods as described in Section 3. Figure 2 uses the rank of the target document as the abscissa. In Figure 3 we show the results using the economically more meaningful measure $w =$[cumulated total number of items examined]. We show only the performance achieved by each team on the 20% degraded materials. Conveniently, almost all team's better effort dominated its weaker effort, in the sense described above. (The exception is ETH, for which the poorer scheme does eventually surpass the better scheme, measured against the work involved.) In Figure 4 we show only the results for each team's better effort. ANU is not represented because this group did not submit a report for the 20% degraded material. Rutgers submitted only one report, as shown here.

Study of Figure 4 reveals that, first of all, the simple pattern matching scheme (*rutcf1*) does most poorly. About 20 target items are found in the first 1200 or so examined, and after that progress is minimal. Initially *CLCON20* and *gmu96v21*, with radically different philosophies, perform about equally. But at somewhere around the 2000th item examined, the n-gram based method continues its slow climb, while the single term assignment method begins to level off.

The *ETHD20P* method, which permits multiple interpretations of a slot in the document, climbs early to a striking advantage, and nearly dominates the other methods. However, there is a small regime, corresponding to the recovery of some 4 or 5 documents near the end of the run, where the n-gram method briefly pulls above the multiple interpretation method. The difference in this region is probably not statistically significant, but it does eliminate the possibility of a clean dominance ordering being reported from this particular trial.

Note that in preparing these figures, we have followed the TREC philosophy that one good document is as good as another, and have not considered which specific target documents were turning up at each particular point on the curve. There is no barrier in principle to doing this analysis, which might prove interesting. In particular, if some schemes do well on some targets, and others do well on others, and *it is possible to tell them apart prior to retrieval*, specific assignment of schemes could produce better performance. Even if it is not possible to tell them apart, some variant of data fusion might produce results superior to those achieved by single schemes.

## 5 Summary

Most teams reported one or more counter-intuitive results at the conference, some of which may have been clarified by the time of the final paper which appears in this bound volume.

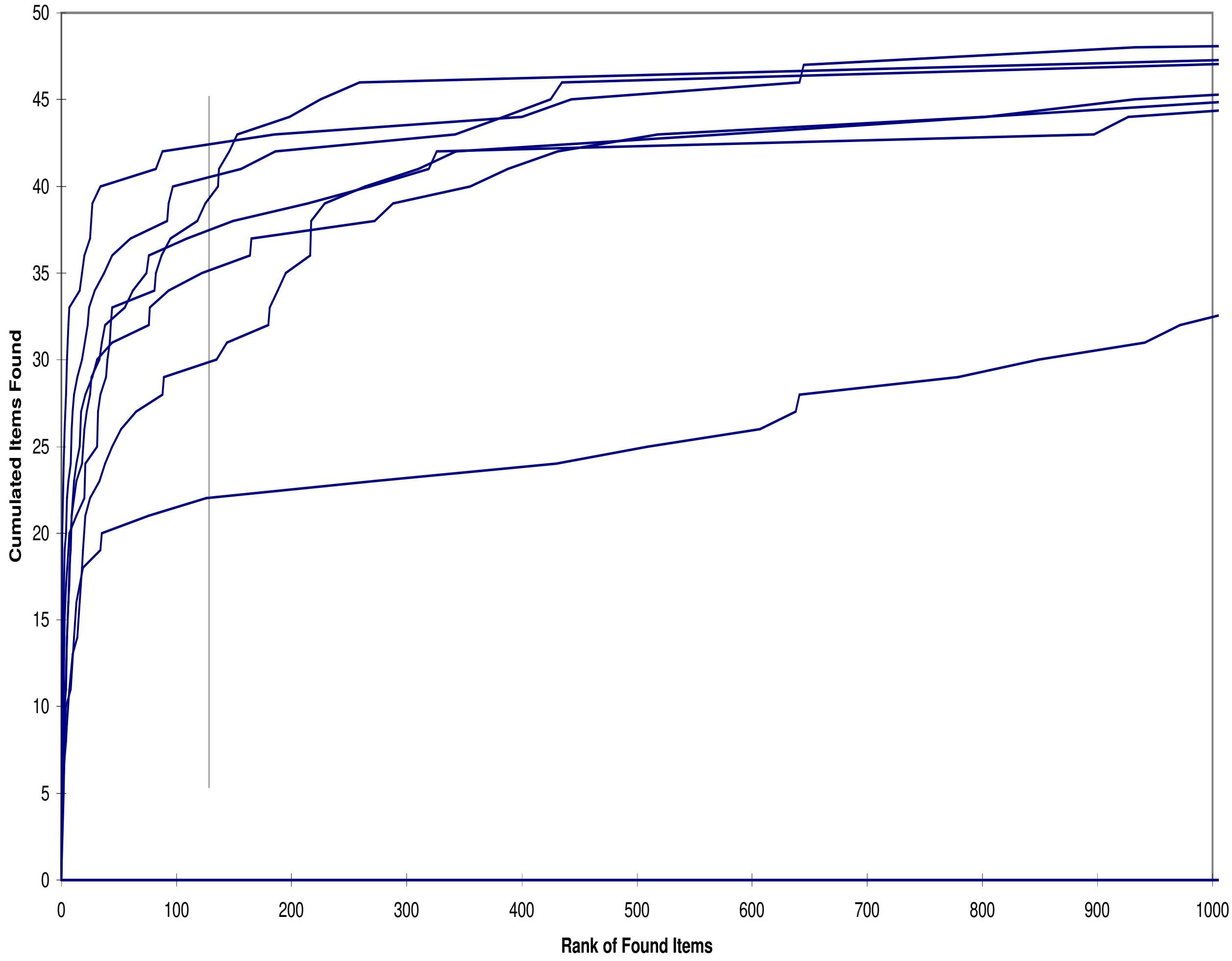


Figure 2: Cumulated relevant documents as a function of the rank in the lists of documents examined. Data are for 20% corrupted text. The systems can be traced by their intersections with the vertical line. In descending order they are: ETHD20P; ETHD20N; gmu96v21; CLCON20; CLCON20F; gmu96v22; rutcf1

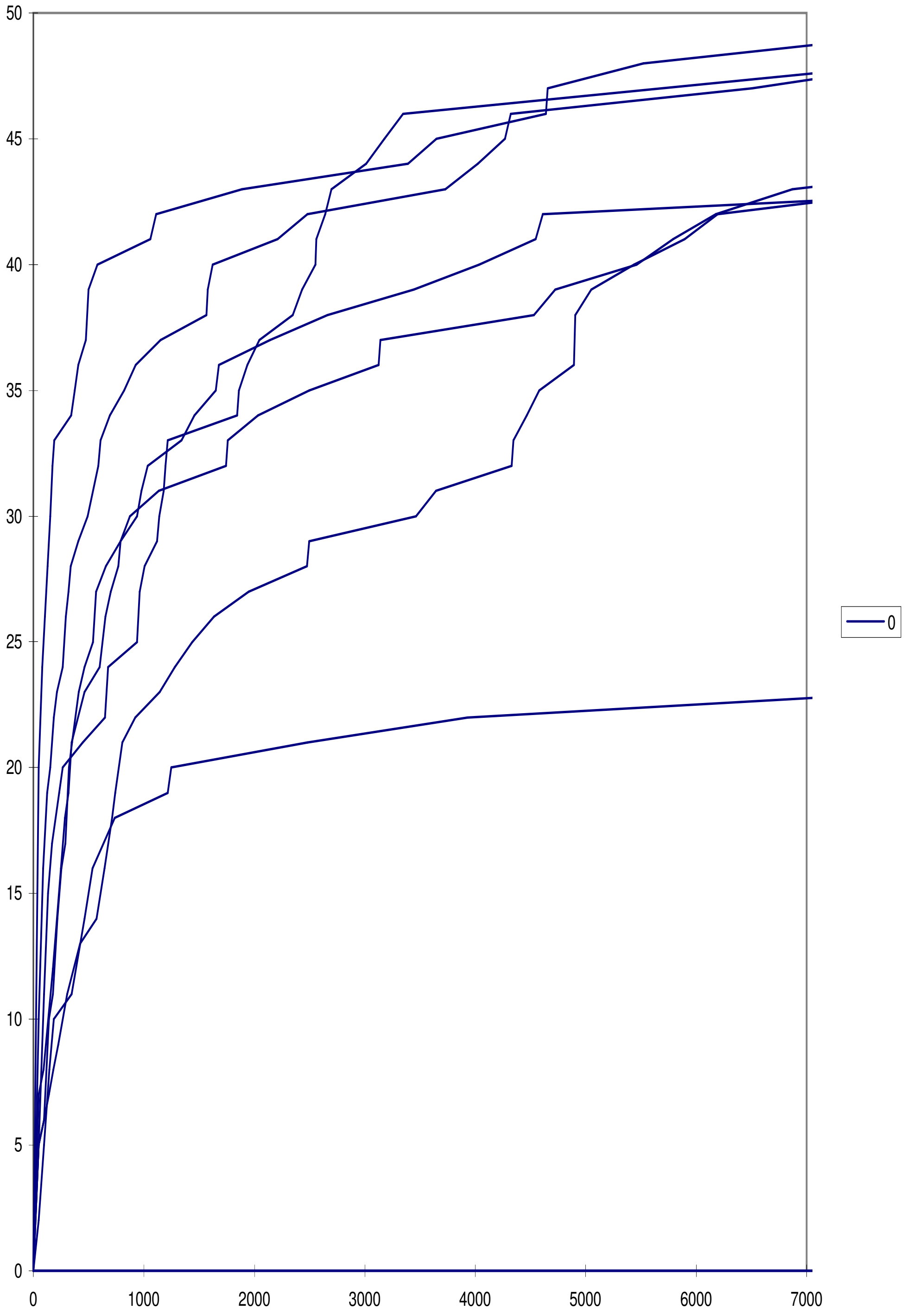


Figure 3: Cumulated relevant documents as a function of the total number of documents examined. Seven systems are shown. Note that the crossing of *gmu96v21* and *ETHD20P* covers a much shorter range when presented in this way. Data are for 20% corrupted text.

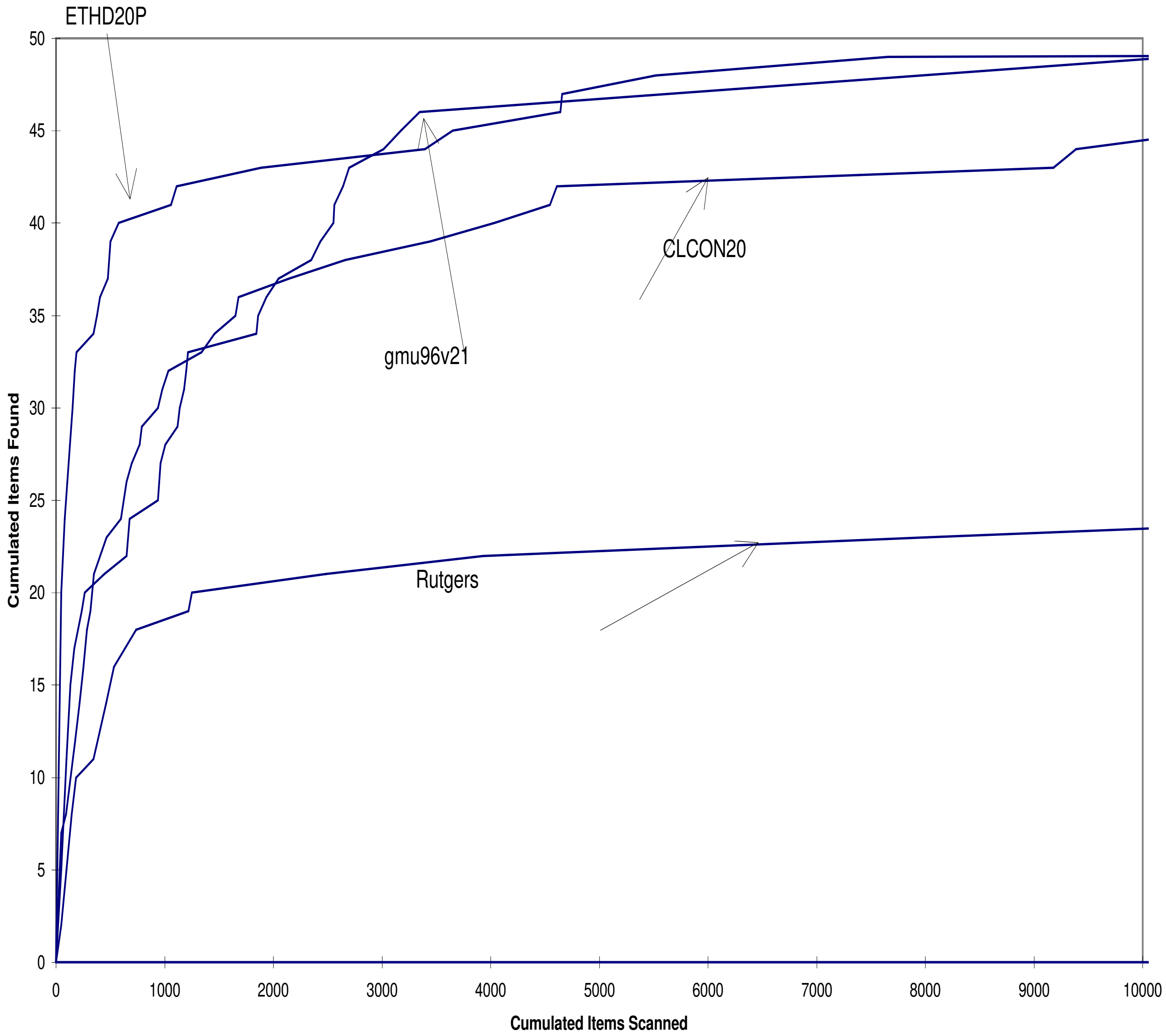


Figure 4: Cumulated relevant documents as a function of the total number of documents examined. Only the dominant best system from each group is shown. Data are for 20% corrupted text.

For example, the ETH initial screening system using 3-grams with Lnu.ltu weighting was substantially more effective than the GMU scheme using 4-grams and cosine weighting. It is not known how much the choice of matching function contributes to this difference, and how much is due to the length of the n-grams. Both the CLARIT team and the GMU team found that their (different) methods of query expansion did not help at all, and made performance worse.

Based on these issues, and the problems noted in the workshop papers, there is still a great deal to be understood about interaction of the diverse approaches used by the participants.

# References

[1] Paul B. Kantor. Non-liner utility functions in information retrieval. Technical Report APLab Technical Report, SCILS, Rutgers University, 1997.