Two Experiments on Retrieval With Corrupted Data and Clean Queries in the TREC4 Adhoc Task Environment: Data Fusion and Pattern Scanning.

Kwong Bor Ng and Paul B. Kantor
APLab, SCILS, Rutgers
The State University of New Jersey
4 Huntington St.
New Brunswick NJ 08903
Internet: {kng@zodiac, kantorp@cs}.rutgers.edu

Abstract

We report on several experiments in using data fusion to improve information retrieval, and in approximate text and 5-gram mathcing methods for retrieval of corrupted text, in the TREC context.

1. Data and Systems

We report on two experiments here. Our plan was to do all experiments on the whole set of TREC4 data, using an IR program called MG (version 1.0, Witten, Moffat and Bell, 1994) as our indexing and searching engine for the experiment. This plan was frustrated (twice) and the official TREC4 submission used a hand-built scanning retrieval technique for dealing with corrupted data.

The reasons for failure of the original approach are lie in the lack of sufficient processing space to manage the file of 5-grams (generated from the 2 Gigabtyes of data files) without reprogramming ; (2) the n-grams approach produced too many distinct terms for MG. Therefore, we decided to conduct both the experiments on only the data of the corruption track.

There are three sets of data for the corruption track: one set of clean data and two sets of corrupted data (10% and 20% corrupted). We conducted two experiments on parts of these data, using different indexing and searching systems. The first experiment was about data fusion. We used the clean set of data as the collection file and MG as our indexing and searching engine.

We originally intended to implement 5-gram searching by using an available very fast indexing and query system, MG (Witten, Moffat, Bell 1994).

The idea was:

> Convert the text to overlapping 5-grams
> Separate all 5 gram-s by NewLine markers
> Submit the resulting text to the MG indexer
> Transform queries in the same way
> Run the queries against the index, using the MG query software.

At the time of these runs we had 9GB of hard disk storage available, partitioned into 1, 3 and 5GB. In general, the 5-grammed version of a text is between 5 and 6 times as long as the text. Since the MG index builder requires a single input file, we had to pipeline the result of the 5-gram expansion into the indexer. However, in its present form, the MG software must build a file of the entire text, for retrieval. Thus we still could not fit the result of the of 5-gram indexing process for the full TREC4 collection into our available storage.

We therefore switched to the Category B corruption track activity alone. This smaller file (less than 0.5 GB) can be managed within the resource constraints mentioned above.

The second experiment explored a mechanism for

retrieving corrupted texts. There was no data fusion involved. We used a pure scan-match-and-count program to test the effectiveness of using "dot-5-grams" (see below) to retrieve corrupted documents.

2. Experiment 1. Data Fusion of the Outputs From The Clean Data

For the experiment on data fusion, we used two different methods to retrieve documents from the clean data. The first method was a ranking retrieval method with word frequency statistics and the cosine rule applied to document vector and query vector to estimate their similarity. We call this the "word-based approach". The second method employed the same strategy except that instead of using the words directly, we used overlapping 5-grams to represent the collection text and the query text (see below). We call this the "5-gram-based approach".

We used the program MG (version 1.0, running on Unix) as our indexing and searching engine. MG employs a csh script "mgbuild" that executes the appropriate sub-programs in the correct order to build a searchable database. We edited the mgbuild into two versions, one for the word based approach and the other for the n-gram-based approach.

2.1 Word Based Retrieval on the Category B Data. Adhoc Queries.

The data for the corrupt track include the WSJ and SJMN files from Disk 2 and Disk 3, in compressed form. We decompressed and concatenated them into one large text file (we called it the clear collection file, 475 Mbytes), then piped it to the first version of the modified mgbuild which accepts a text file piped in as input to construct the MG database. During indexing, each document is assigned an MG document number. We maintained a dictionary program external to MG to translate MG document numbers to original document numbers.

MG uses a sub-program "mgquery" to do the searching. We divided our queries expansion into two steps: preliminary and refined. In the first step, we used the 50 topics as queries and retrieved the top 50 documents (i.e., with highest cosine scores) for each topic. These could serve as expanded queries. We read the output to identify appropriate documents and used them as the refined expanded queries for the second round of search. For topics 223, 233, 245, and 246, we found no appropriate documents so the refined expanded queries were the same as the preliminary queries. For the other topics, we successfully identified some number of relevant documents (1 to 8) to serve as the refined expanded queries.

In the second step, we used a nawk script to pipe into mgquery the 50 refined expanded queries and, for each topic, get the MG document numbers and the relevance scores of the top 1000 documents output by MG. We call this output "clean words output".

2.2 N-gram Based Approach

2.2.1 Formation of 5-grams.

In this approach, we decomposed the collection texts and the query texts into their constituent n-grams and used the n-grams instead of the words to do the indexing and searching. Since n-grams of length 5 balance computational time and performance (Damashek, 1994), we used 5-grams. Before decomposing the texts to 5-grams, we translated all punctuation to period (.), and we escaped upper case letters to slash (\) followed by the corresponding lower case letter. All white space (tab, space, newline and the other non-alphanumeric codes) were translated to tilde (~). For example, before indexing or searching, the text:

Pear?
No. Orange!

is translated to

\pear.~\no.~\orange.~

The text is then decomposed into overlapping 5-grams. For simplicity each 5-gram is placed on a new line. (This wastes paper, but not disk space).

/pear
pear.
ear.~
ar.~/
r.~/n
.~/no
~/no.
/no.~
no.~/
o.~/o
.~/or
~/ora
/oran
orang
range
ange.
nge.~

2.2.2 Construction of the Database and Index

We piped the collection file to our second version of the modified mgbuild script, which contained our own programs (sed and C, by C. Basu) to do the translation as described above. The output is called the 5-grams database. The relationship between the MG document numbers of the 5-grams documents and the original document numbers in the collection file remains the same as in the word-based approach, and we use the same dictionary program for translation between them.

2.2.3 Retrieval

We wrote another nawk script to do the retrieval. It translated and decomposed the texts of the 50 topics into the required 5-gram format, piped them to mgquery, searched the 5-grams database, and for each topic found the MG document numbers and relevance scores of the top 1000 documents output by MG. We call the output "clean 5-grams output".

2.3 Data Fusion

The preceeding work results in two lists of the top 1000 documents, for each Topic in the Adhoc set (201-250). One list ("words") contains the top 100 documents resulting from the word based search, together with their relevance scores (calculated by a cosine method). The other contains the scores obtained when searching for the 5-grams of the query, in the 5-grams of the documents themselves. For each Topic the scores were normalized to lie in the interval [0,1], with the top ranked document assigned score 1, and the 1000th document assigned score 0. This maintains the relative size of gaps between documents.

We considered two schemes for combining the scores. One, called "mean" assigns to each document the mean of the scores which it has in the two lists. This approach is rigorously justified if (and only if) the odds that a document is relevant is a function only of this mean. This will occur if the odds are exponential in the mean, and if the distribution of documents in the space represented by the two scoring methods (words and 5-grams) has a simple product form [Kantor, 1995]. Methods of this type have been used with some success in earlier work by our own group and by others. [Belkin et al; 1993, Belkin, Kantor, Fox, Shaw, 1994]. The general notion of adding scores can also be supported by arguments about the scatter of retrieval methods about a true center, in a vector space approach [Kantor, 1994a].

The second scheme for data fusion implements an idea which can be stated easily in words [Kantor, 1994a], and which found some (very thin) support in our work in TREC3 [Kantor, 1994b]. The idea is: "if two schemes for retrieving documents are different, but each has some sense to it, then they are more likely to agree that relevant documents are relevant, than to agree that non-relevant documents are relevant." This has worked rather well with schemes which correspond to indiviudal human beings, when the notion of "agreement" was interpreted as overlap of Boolean retrieved sets. [Saracevic and Kantor]. In the present setting, with ranked sets, there are several possible interpretations of the Boolean overlap [Kantor, 1995; Belkin et al, 1994]. We chose to explore the variance of the rescaled normalized scores as such a measure.

Thus, from the two sets of ranked lists (words, 5-grams) for each topic, we produce two data fusion lists (mean, variance). Since not every item appears on both lists to be fused, we adopted the further rule that all items appearing on both lists are ranked above items appearing on only one list. All items on both lists were ranked in decreasing order of the mean score (for the mean fusion), and in increasing order of the variance of the two scores (for the variance fusion). These two official submissions for the uncorrupted data were labelled rutfum (rut=Rutgers; fu=fusion; m=mean) and rutfuv (v=variance).

3. Experiment 2. Corrupted Data

For the two sets of corrupted data (10% and 20% corrupted), at first, we planned to employ the same scheme as in Experiment 1. For example, to generate the "words" list we would use MG as the indexing and searching engine, with the same refined expanded queries as before. But the database would be built from the corrupted versions of the documents. However, the corrupted texts contain so many distinct words that we could not use MG to handle the indexing. We therefore abandoned, with regret, the fast indexing and searching capabilities of MG, and developed our own full scanning retrieval "engine". Scanning has been reported in the TREC environment previously. [Cavner; Mettler and Nordby]

Our scanning method is a modified version of the n-grams approach, motivated by the following considerations.

The process of translating and decomposing text to n-grams may generate at least n times as many terms (n-grams) for indexing. The "strangeness" of the spelling of the n-grams would also make ordinary stemming technique very difficult to apply. In addition, since the texts were corrupted, there were a lot of unusual words, and possibly some strange codes, that would make the inverted index file much larger than normal. In fact, we experienced some problems in attempting to use MG to construct the corrupted 5-grams databases. Since the purpose of this experiment is to test the mechanism, not the speed, we wrote our own scripts (csh, nawk, and perl) to do the searching and retrieval instead of using MG.

In corrupted texts, the spaces separating words might not be true correct word-boundaries as in normal text. In addition, any character of the text could be a corrupted space. This might cause a decrease in matching for the corrupted texts if we treated the spaces as word-separators and treated the non-space as non-word-separators. The use of dotting (described below) provides some measure of protection against this.

On the other hand, since the query texts were not corrupted, using spaces as word separators might compensate for the effect of the possible corruption of this function in the corrupted texts.

Our scanning process is enormously slower than the full indexing using MG. We were forced to remove all stop-words in the original topics before generating the 5-grams queries. This destroys semantic and syntactic relationships which would otherwise be parially represented by the overlapping 5-grams. With the queries reduced to bundles of non-stop-words, it does not make much sense to generate 5-grams across word-separators, and we did not. If the words had 5 characters or less we simply kept them untouched (for the sake of simplicity, in this paper we still call them 5-grams). For example, the 5-grams generated from the question:

What is a pear and what is an orange?

would be:

pear
orang
range

3.1 Dotted 5-grams

After getting the 5-grams from the topics, we generated from them what we called dot-5-grams. That means, for each character in the 5-grams, we generated the corresponding dot-5-gram by replacing that character with a dot. For example, the 5-grams "orang" would create 5 dot-5-gram, i.e., ".rang", "o.ang", "or.ng", "ora.g", and "oran.". Later in the process of matching, the dot would match with any character or code in the text.

The mechanism of the retrieval was simply the match and count mechanism, without weight consideration or vector calculation. We compared the dot-5-grams generated from each topic with the corrupted texts and counted the number of matches. Then we retrieved the top 1000 documents with the highest counts for each topic. These were then resorted in decreasing order of the ratio of the number of hits to the number of lines in the document (hits per line , or hpl). We call the outputs "10% corrupted dot-5-gram output" and "20% corrupted dot-5-gram output" respectively.

3.2 Dotted 5-gram retrieval

The pseudo-code for this dotted-5-gram retrieval is shown in Exhibit 1.

Exhibit 1. The dotted-5-gram algorithm

```
for each document
  hits=0
  for each line in the document
    for each 5-gram in the query
      for each dotting of the 5-gram
        if dotted-5-gram matches line
          hits=hits+1
        end if
      next dotting
    next 5-gram
  next line
  print document sequence number,
  print hits
next document
Sort documents by hits per line
```

The specific steps of this algorithm were implemented in nawk scripts, which run very slowly (typically 8 to 12 hours to complete a query with 25 words). To complete the analysis in time for this conference, the queries were

reduced to content words alone, as described above.

3.3 Chance of missing a term

The rationale for this approach is illustrated by a calculation of the chance that a given term of length c will be missed by this algorithm, or will result in a specified number of hits h.

If there is no corruption, a term of length c will have h=1 hits if (c<6). If (c>5) then the number of hits will be equal to the number of overlapping 5-grams contained in the term, or h=c-4. For example, a term with 6 characters will have two overlapping 5-grams, both of which hit it.

In the presence of corruption at a level e, any of the characters has a probability e of being replaced by some other character. Disregarding, for the moment, the problem of false word boundaries, we see that the chance that all 5-characters in a particular position of the 5-gram window are intact is reduced to $(1-e)^5$.

The chance to have exactly h hits, if the term is of length c, is given by a more complex calculation, in which the number of times that each character is covered by a 5-gram is taken into account. For example, if the character in the middle of a nine character term is corrupted, then none of the 5-grams will hit. But if the first character is corrupted, then there will be four hits as the 5-gram window moves across the term. Details will be given elsewhere.

In sum, if a particular 5-gram is converted to all 5 possible dottings we improve robustness against corruption of the characters in the target term.

4. Performance Results and Discussion

4.1 Performance of Data Fusion by Mean Score. Clear Texts.

We look first at the results on the clear data. Here we used queries which were, in some cases, substantially expanded into a set of as many as 8 documents. The results are substantially better (we consider here only the mean fusion procedure). The results are not, however, at or above the median. The results for those topics for which we did not score below the median are summarized in Table 1.

Topic	OurScore	BestScore	Median
202	63	67	58
209	13	15	10
211	62	62	25
215	47	47	32
216	19	19	12
219	21	21	12
221	43	43	25
224	18	26	13
226	27	27	9
227	30	33	25
234	4	4	3
238	24	28	8
239	6	6	5
240	37	37	21
243	17	17	10
245	8	8	5
248	7	7	2
250	14	14	11

Table 1: best scores, our scores, and median of data fusion applied to clean texts. Results for the mean method of fusion are shown.

We are pleased to note that on quite a few of these our score is equal to the best score. But overall, we cannot say that the results of this scheme are "above the median". More precisely, with 95 percent confidence, they are below the median.

4.2 Performance of Dotted 5-grams and Scanning methods.

Although we have done them by different methods, we can compare our performance with that of the other systems, as reported by NIST. For 20% corruption the results are indeed dismal. Our results (measured in the number of relevant documents at 100 retrieved documents) was equal to the lowest of all reported scores in all but 7 of the 50 cases. Since there are only two entrants in this category, we have come in a poor second. Is there any reason for hope?

An optimist would point out that the n-gram and dotting

techniques are expected to take advantage of having large queries. Thus we should not expect good results with sharply abbreviated queries, often stripped down to three or four words. In this sense, what we have done so far is simply establish that we can perform all the calculations called for, but not in a particularly efficient manner. Clearly a great deal remains to be learned. The specific topics on which our results were not the worst are 205, 208, 220, 224, 228, 229, 239 and 240.

5. Conclusions

5.1 Data Fusion: Results and interpretation.

The results of our data fusion experiments are summarized in Tables 1,2 and 3. We see that the results of fusion are not significantly better than the results of the two input schemes: term-based and 5-gram based. The results of fusion based on variance are too poor to be discussed. For the fusion based on mean scores we will try to understand why the fusion process did not score any better than shown. We will not enter into a case-by-case analysis. A preliminary examination of the scatter plots for the four cases in which (a) fusion was somewhat better than either of the input schemes and (b) there were more than a few relevant documents in the top 100 did not reveal any informative patterns.

Scheme	Avgp100	N (Topics)
fuv	0.050	49
fum	0.144	49
c20	0.041	49
c10	0.047	49
c0	0.050	49
word	0.149	49
5gram	0.138	49

Table 2. Average, over all Topics of the Precision at 100 documents.

Data Fusion Schemes and Input Schemes				
	fuv	fum	word	5gram
fuv	0	3	3	4
fum	39	0	19	17
word	39	10	0	15
5gram	39	10	19	0

Table 3. Comparison of two data fusion schemes with the performance of the two input schemes from which they are formed. The entry in each cell is the number of times the scheme labelling the row performed better than the scheme labelling the column. All comparisons are based on the number of judged relevant documents retrieved in the top 100 positions.

To show what kinds of patterns are sought, we plot (Figure 1) the scores for the top 100 documents, after mean fusion, showing the normalized scores based on normalization over the top 1000 documents. This permits us to show points representing documents for all of the topics on the same graph. The significance of scoring suing the term rule is that a vertical line sweeps across the graph from right to left, and documents are included as the line passes them. For the 5-gram rule the line sweeps down the plot, from the top, and documents are included as the line passes them. Finally, for the mean rule (which is logically equivalent to a sum of scores with equal weights) the line makes a slope of 45° with the axes. There are some interesting features of this plot to which we return below.

In Figure 2 we show a "zoomed" view of the sub-plot for documents both of whose normalized scores fall in the interval [0.2-0.5]. Examining Figure 2 we see that when all three schemes have just picked up the document which is represented by the point at the intersection of the three lines, the quadrant above and to the right of the point represents documents which have been included under all three schemes. The triangle on the lower right (a) includes points corresponding to documents which are retrieved under the f-gram scheme, and not under the mean fusion. On the other hand, the point in the trapezoid on the upper left (A) are included under the mean fusion scheme, but not included under the 5-gram scheme. Thus the mean fusion will be better than the 5-gram scheme if and only if regions of type A generally contain a higher ratio of relevant to not relevant documents than do the corresponding regions of type a. This relationship can be explored systematically. For the present we simply note that the graphic representation does not suggest, to the eye, any such preference for the mean fusion scheme.

Returning now to the plot of Figure 1, we consider the overall structure. The points are quite concentrated along the diagonal. This means that the two schemes (terms and 5-grams) are not providing independent information about the retrievable documents. This is precisely the situation in which data fusion will not be expect to work. In the limit, if the points lie on a single line (even a curved line) there is no advantage to knowing both coordinates of a point, as either determines its position in the overall ranked set.

For the most part, the points far from the principal diagonal are dots ("."), representing non-relevant documents. This

is to be expected,a nd is summed up by the proposition that "good retrieval schemes are more likely to disagree about the score assigned to non-relevant documents than about the score assigned to relevant documents". This is the motivation, as noted above, for the fusion scheme based on variance. However, for the two schemes combined here, the fusion based on variance does extremely poorly. This is precisely because nearly all the documents, whether or not they are relevant are clustered close to the principal diagonal (which corresponds to variance 0).

We finally note a peculiar feature of Figure 1, which is the region in the lower right which is far from the central diagonal, but is occupied entirely by relevant documents. This is exactly the kind of phenomenon postulated when data fusion is viewed as a form of adaptive pattern recognition in the space of multiple attributes. Of course this region is far too sparsely occupied for us to apply that method here, but it serves to illustrate the concept. Mathematically this region is defined by high values of the difference *(x-y)*, where *x* represents the term score of the document. In fact, selecting the cases for which *(x-y)> 0.4* we find that 10 of these cases come from topic 223, and ten of them come from topic 225. The significance of this is not yet understood. It appears that for these topics the difference of scores might be an effective rule of combination. But this conclusion may be the equivalent of statistical data mining, and not significant.

5.2 Scanning and Confusion

Turning now to the retrieval on corrupted data, we show, in Tables 1 and 3 the results for retrieval using the scanning scheme on uncorrupted, 10% corrupted, and 20% corrupted text. These schemes degrade as expected. However, they perform poorly in comparison to schemes which index all the 5-grams, and use the original texts as queries. Recall that our scanning scheme uses severely truncated queries in order to control processing time.

Corrupted Data. Scanning schemes			
	c20	c10	c0
c20	0	5	5
c10	17	0	7
c0	26	17	0

Table 4. Comparison of performance of the scanning scheme on data at different levels of corruption. The entry in each cell is the number of cases in which the scheme of the row scored better than the scheme of the column.

5.3 Overall Conclusions

Discussion and Prospects.

We find two principal results. First, the 5-gram indexing scheme and the term-based indexing scheme (which includes a stop list) seem to provide scores for each document which are too similar to support a priori fusion schemes based on the variance, the mean, or a weighted sum. While we present here only the scatter plots for all 49 topics together, the conclusion is much the same for the individual topics as well. Second, the weak-match scanning scheme used with very short queries does not handle corruption as well as the several indexing schemes reported by other participants [Buckley et al; Frieder et al; Huffman ; this volume]. Experiments are planned to explore the dependence of this performance on three possible improvements: (1) use of an "inverse document frequency" weight to decrease the importance of patterns which are matched frequently (2) the use of a center of gravity transform as an alternate method for dealing with the issue of document frequency (3) extension to longer queries, based on relevant retrieved documents.

It is clear that the two methods reported here (data fusion by mean score, for the clear texts, using refined expanded queries composed of zero to eight relevant documents; and scanning with key-word queries and dotted 5-grams) do not advance the state of the art, as represented (at a minimum) by the current TREC entrants. Without regard to the state of the art, fusion by minimum variance of the normalized score performs dramatically more poorly than fusion by the mean. For unfunded research, and newly developed systems, we feel that these results are not embarassing. In particular, a number of possibilities are open for further exploration. They include:

1. Revisiting the original (word and 5-gram) lists which entered into the data fusion. Do either of these lists perform significantly better than fusion by the mean, which is our best official entrant.

2. Examining alternative fusion schemes. The symmetric sum, represented by the mean, and the variance, represent only two among a continuum of possible rules for data fusion.

3. Improving the completeness of the treatment of corrupted texts. Since results, in comparison to the median, are much better for clear texts than for corrupted texts, we might, as noted above do better on the corrupted texts by expanding the queries and/or doing some variant of data fusion.

Finally, the MG program which we applied to the clear text supports important weighting and stemming features whose analogues for the corrupted case are not apparent. In general, our design philosophy is to avoid language-specific devices (morphologies, externally generated thesauri, etc.) but there may be some analogues to the corpus-generated thesaurus which are appropriate for the case of corrupted data.

6. The Authors

K.B. Ng is a graduate student in the PhD program in Communication, Information and Library Studies at Rutgers, and holds a teaching assistantship there. P. B. Kantor is Professor in the School of Communication, Information and Library Studies at Rutgers, where he directs the APLab, and is also a Member of the Center for Operations Research, and a Faculty Associate of the Project in High Performance Computing and Design.

7. Acknowledgements

We are, in the words of T. Williams, dependent upon the kindness of others again this year. First to the team in Melbourne, for creating and making available the MG engine, and for providing support at odd hours of the day and night as we tried to bend it to tasks for which it was never intended. Particularly to Tim Sinnott, technical support. At Rutgers, Chumki Basu, a graduate student in computer science wrote several filters in C, and also scanned the great majority of the retrieved documents in order to build the refined enhanced queries. Oleg Goldenstein, a graduate student in Library and Information Studies joined the project late, and worked on a compiled version of the scanner, which has not yet been implemented. As always, intra-departmental rivalry with the team headed by N. Belkin, of Library and Information Studies, working on relevance feedback and interaction was a constant and pleasurable stimulus. The support of Dean Richard Budd for the Alexandria Project Laboratory (APLab) has been essential in the development of the necessary infrastructure. The APLab is also supported by grants and contracts from private and federal institutions.

8. References:

Damashek, M. (1994) Gauging Similarity via N-Grams: Language-Independent Sorting, Categorization, and Retrieval of Text, Science, 267, (10 Feb., 1995) 843-848.

Belkin, Kantor, Fox and Shaw. Combining Evidence for Information Retrieval. in D. Harman, Ed. Proceedings of the 2nd annual Text Retrieval Conference. NIST 1994. pp35-44

Belkin, Kantor, Cool and Quatrain. (1995). Combining the Evidence of Multiple Query Representations for Information Retrieval. Information Processing and Management 31(3)431-448.

Cavnar, William B. N-Gram-Based Text Filtering for TREC-2. D. Harman, Ed. Proceedings of the 2nd annual Text Retrieval Conference. NIST 1994. pp171-179

Kantor, Paul B. Data Fusion in Information Retrieval: Towards a Theoretical Foundation A: Vector Simulation Models.. APLab Technical Report. APLab/TR-93/3. Rev. Sept. 1994a.

Kantor, Paul B. Decision Level Data Fusion for Routing of Documents inthe TREC3 Context: A Best Case Analysis of Worst Case Results. in D. Harman, Ed. Proceedings of the 3nd annual Text Retrieval Conference. NIST 1995. pp

Kantor, Paul B. Tutorial on Data Fusion in Information Retreival (1995). ACM SIGIR.

Mettler, Matt, Nordby, Fritz. TREC-II Routing Experiments with the TRW/Paracel Fast Data Finder. in D. Harman, Ed. Proceedings of the 2nd annual Text Retrieval Conference. NIST 1995. pp319-332..

Saracevic, T, Kantor, PB, A Study of Information Seeking and Retrieving. III Searchers, Searches, Overlap. JASIS.39:178-194 (1988)

Witten, I. H., Moffat, A., and Bell, T. C. (1994) Managing Gigabytes. New York: Van Nostrand Reinhold.

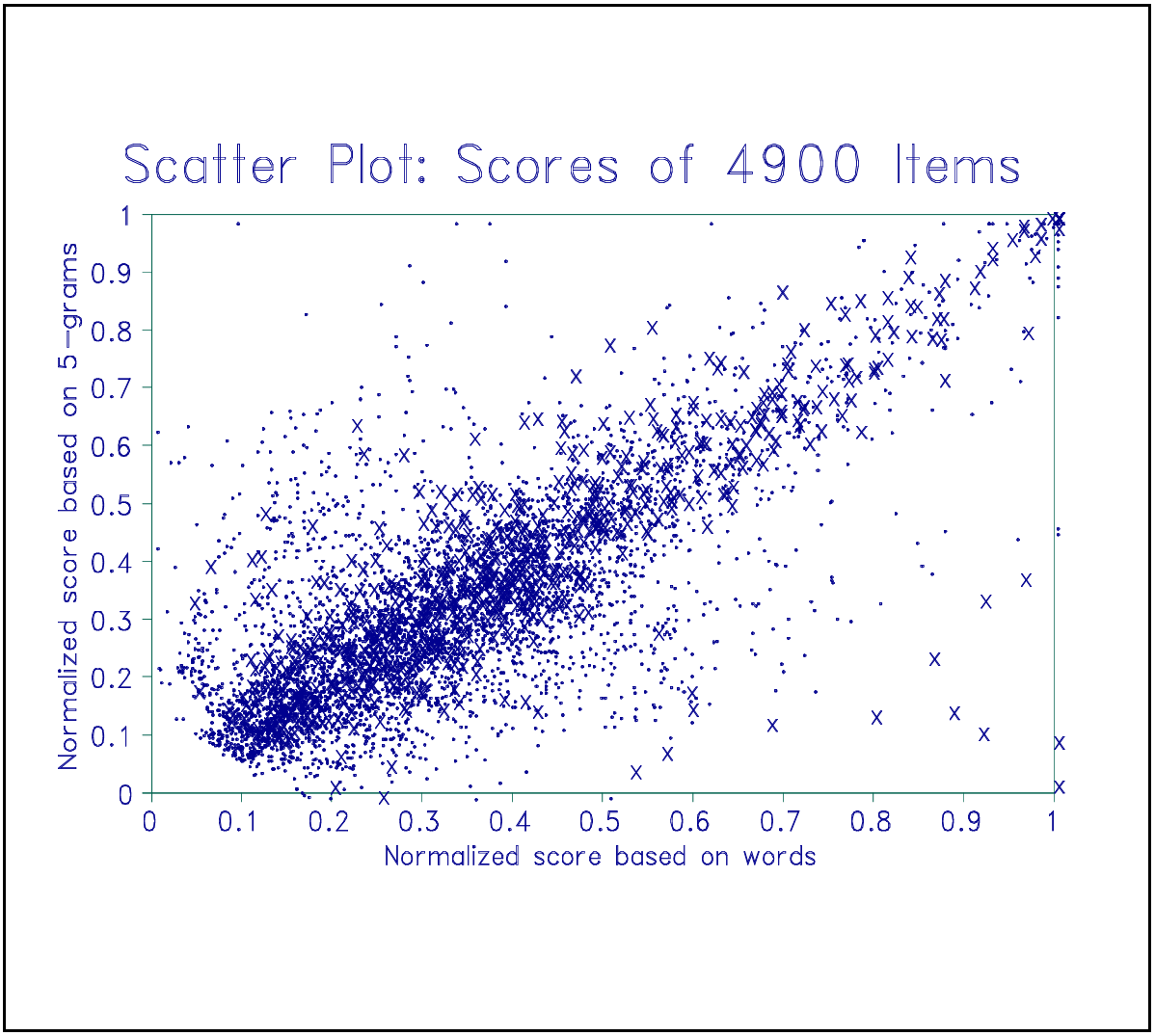

Figure 1. Scatter plot of the scores assigned to the top 100 documents for each of 49 adhoc topics, "x" represents a relevant document and "." represents a non-relevant document.

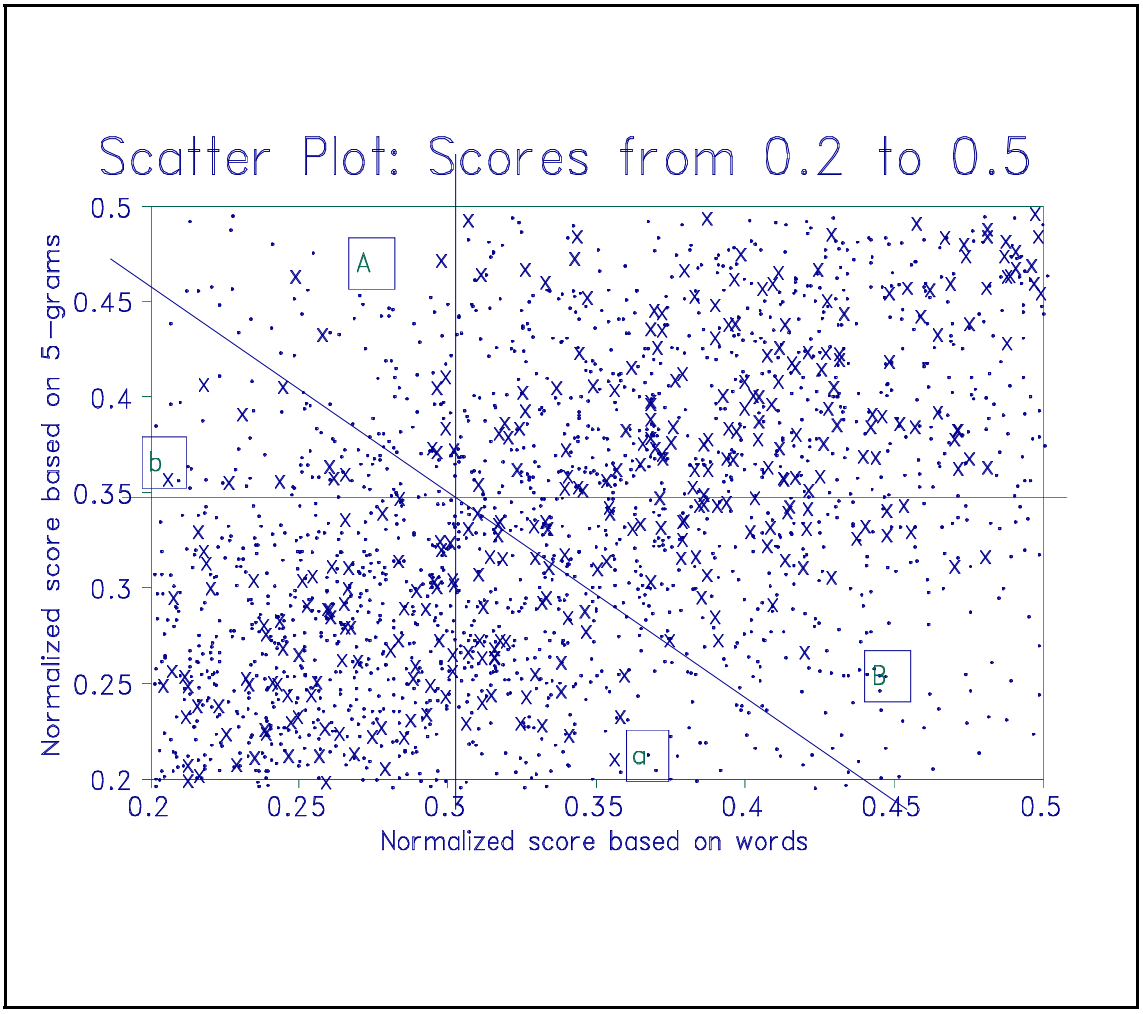

Figure 2. Enlarged portion of the scatter plot, showing level curves for term-based (words), 5-grams and mean. Graphic errors have changed to diagonal line, which should pass through the point 0.45 on each axis.