

Document Retrieval In OCR-Scanned Text

David Hawking
Co-operative Research Centre For Advanced Computational Systems
Department Of Computer Science
Australian National University
dave@cs.anu.edu.au

February 19, 2007

Abstract

The use of a document retrieval system (PADRE) for the Fujitsu AP1000 in processing known-item search queries over OCR-scanned documents is reported. Retrieval performance of an initial set of queries is shown to deteriorate significantly over scanned data with a character error rate of 5%. A preprocessor is used to augment queries with terms which can be derived from original terms using characteristic substitutions observed to occur in a sample of the scanned text. This technique is shown to markedly improve performance over the degraded data.

1 Introduction

Document retrieval systems are now in widespread use by ordinary individuals as well as by employees of corporations and departments. Consequently, search services on the World-Wide Web now process huge numbers of requests. At the time of writing the Alta Vista server reportedly carries out twelve million searches per day.

A serious restriction on the utility of retrieval systems is the gulf between information in electronic form and that existing only on paper. Users who appreciate the enormous labour saving potential of electronic searching are frustrated because still-relevant older data cannot be searched.

Optical Character Recognition technology (OCR) may form part of a solution to this problem but error rates when scanning may be high

if the printed originals are not in ideal form. Error rates increase significantly when originals are noisy or written in fonts for which the OCR software has not been trained. Major structural errors may also occur as a result of *zoning* errors in which the regions of text to be scanned are incorrectly identified. For example, multiple columns may be scanned as though they were one and figures, borders and photos may be scanned as text. Zoning errors are not considered here.

Human correction of the errors in large collections of documents is too expensive to consider in most large-scale applications.

Fully automatic correction using spelling-checkers is not likely to be broadly beneficial because spelling-checkers rely on the use of a dictionary of valid words. In general, collections of documents will contain huge numbers of words which are not found in regular dictionaries. Such words may be acronyms, proper names, foreign language words, esoteric technical terms and archaic locutions. It has been observed [1] that there are over 700,000 distinct words in 2 gigabytes of text distributed as part of the TREC collection. [4] Even above the 2 gigabyte size, a collection lexicon does not remain static and it is estimated that one novel word will occur for each thousand added.

An automatic spelling checker will fail:

1. if the original word is not in its dictionary,
2. if the OCR substitutions resulted in valid but incorrect words, or
3. if the garbled word is an equal "distance" from multiple valid words.

Is it possible to devise search engine technology which can operate effectively over uncorrected OCR-scanned data?

The United States National Institute of Standards and Technology (NIST) are this year sponsoring a trial of document retrieval over OCR-scanned data as part of the TREC Conference [4]. A collection of several hundred megabytes of US Federal Register documents is provided in three forms:

1. An error-free “truth” version,
2. an OCR version with an error rate of approximately 5% (degrade5), and
3. a severely degraded OCR version with a 20% error rate (degrade20).

The SGML tags identifying starts, ends and labels of documents were preserved free of corruption.

The particular retrieval problems are *known-item searches*. Staff at NIST, assisted by retrieval software, located one-of-a-kind documents in the “truth” collection and composed natural language specifications of them. Examples of such specifications are as follows:

I'm looking for a record of a request for a grant to a jazz format public radio service for a large city in Louisiana.

What evidence is there that companies which adopt a policy against smoking can lower their maintenance costs?

Education programs for mammographic image quality assurance.

Participants in the trial were required to generate retrieval queries to locate the required item. Results, in the form of a ranked list of supposedly relevant documents, are scored by how early in the list the sought-after item appears.

This paper describes the methods used in the Australian National University participation in this trial using the PADRE document retrieval engine for the Fujitsu AP1000 [6].

1.1 Past Work

Four groups attempted the corrupted data track at TREC4 in 1995: Cornell Uni., Rutgers Uni.,

George Mason Uni. and the U.S. Department of Defense. It is understood that two of these groups [3] used n-gram methods. The Rutgers group [8] also used n-grams but fused the output from the n-gram run with a two-stage word based run which operated as follows: First, the raw topic was used as an initial query then the top 50 documents retrieved were used as an expanded query. The rationale for this approach is that retrieved documents will probably contain corrupted as well as uncorrupted versions of significant terms. Frequently occurring corruptions (due to systematic scanning errors) will be added to the new query via the normal relevance feedback method.

Cornell's approach [2] was rather different. It added to an initial query all words occurring in the text which can be generated from an initial query word (must be longer than 5 characters) using a single character transformation (addition, deletion or substitution). Cornell's normal term weighting methods result in excessively high weights for rare mutations of common words and consequently a second pass was used to ensure that all transformations of the same query word were taken into account in computing term weights.

The Cornell approach appeared to be more successful than the others but it should be noted that the TREC4 task was corrupted using a random garbling algorithm rather than the systematic mis-recognition behaviour typical of OCR-scanned text.

Outside the TREC context, Taghva et al [10] have specifically addressed the retrieval problem over OCR-scanned text belonging to the Nuclear Regulatory Commission. Essentially, they found that retrieval performance was not particularly adversely affected by the presence of scanning errors. They also attempted to use correction by automatic spelling checking against a dictionary derived from their particular collection but results were so good over the original data that little further improvement was gained. Unfortunately, Taghva et al do not specify the scanning error rate nor do they give enough information about the types of query (or document) to determine whether the same observations might apply more generally.

Smith and Stanfill [9] also found that doc-

ument retrieval performance remained surprisingly high despite high scanning error rates. They simulated OCR corruption by randomly replacing characters with a **recognition failed** symbol. Consequently, any new words introduced by the corruption were guaranteed to be substrings of existing words. This is not the case in the TREC corruption task. Smith and Stanfill found that relevance feedback methods were more resistant to corruption than a straightforward Boolean method. This is probably because relevance feedback added substrings of significant words to the augmented query.

2 Choice of Method

In line with the successful Cornell approach described above, it was decided to employ a strategy in which no modifications were made to the retrieval engine itself. Instead, pre-processing was used to augment the original query by adding variants of the query terms which could result if the terms were subject to OCR errors.

Query pre-processing can potentially result in a very large number of query terms with consequent increase in search time. One strategy to minimise this problem is to use PADRE to form a lexicon (a complete list of all distinct words) for the degraded data and to remove augmented query terms which do not actually occur in the data.

The first method considered for solving this problem was to use the approximate pattern matching algorithms of Manber and colleagues at the University of Arizona. [11] In this approach the original query terms would be sought in the lexicon for the corrupted database using the **agrep** program. This program can be used to find all the terms in the lexicon which match an original query term with at most s substitutions, insertions or deletions.

Some problems are soon apparent with this approach. First, it assumes that confusion of visually unrelated characters such as **X** and **o** is just as likely as confusion of visually similar characters such as **l** and **1**. With OCR data this assumption is clearly false and may cause the query to be augmented with large numbers of distracting terms, which increase query processing time and, more importantly, the false hit

rate. Second, the **agrep** program takes a parameter which specifies the maximum allowed number of mismatches. If this parameter is set to a relatively large value, such as three or four, the number of query terms will increase dramatically. If, on the other hand, the parameter is set to one then many actual distortions of a query term will not be matched. For example, if a particular OCR system recognises **s** as **g** then the word **Mississippi** may be scanned as **Miggiggipi**, with four substitutions.

Accordingly, it was decided to observe and classify the transformations occurring in a small sample of the degraded data and to generate additional query terms, using only the observed transliterations. This process generates a large number of alternate terms if it is assumed that the transliterations occur with a probability less than unity. If, in other words, the OCR system sometimes gets it right! If a query term contains three letters which are observed to be subject to transformation, seven (that is $2^3 - 1$) new terms may need to be added, to cope with all possible combinations of particular substitutions occurring and not occurring. The query preprocessor allows the user to select whether substitutions always occur or sometimes occur.

3 Details of Query Pre-Processing

3.1 Replacement of one Alphanumeric Character by Another Character

This is the most commonly occurring type of transformation. It does not result in a change of term length. The preprocessor must treat case sensitive terms differently to others because some transformations merely change the case of the original letter (for example **P** \rightarrow **p**) and sometimes upper and lower case forms of a letter transform to different results. For example **J** \rightarrow **'** but **j** \rightarrow **d**. When processing a **j** in a case insensitive term, three results are possible: **'**, **d** and **j**.

If the substitution results in a rare punctuation character, extra search time may result when using PADRE's Super Dictionary Method.

3.2 Replacement of one Non-Alphanumeric Character by Another

One example of such a transformation was observed: `space` \rightarrow `)`. Although PADRE query terms may contain punctuation characters, they usually do not. However many terms contain spaces, which match any single non-alphanumeric character. Consequently, replacement of one non-alphanumeric character by another due to scanning errors may usually be ignored. This was the case in the experiments reported below, as no query terms contained punctuation marks.

3.3 Replacement of one Alphanumeric Character by a String

Sometimes one character is transformed to a sequence of characters. The only such example detected in the sample was `f` \rightarrow `ff`. The terms resulting from such substitutions may be longer than the original.

3.4 Replacement of Multiple Alphanumeric Characters by a String

The only observed instance of this in the sample was `ffi` \rightarrow `M`. For simplicity, `ffi` was always replaced with either `M` or `ffi` even though `fffi` and `ffffi` may also be possible.

3.5 Replacement of Non-Alphabetic Character by Alphanumeric

These substitutions may require the addition of characters at the beginning of the original term. This is true when a digit is replaced by a letter as PADRE insists that words must start with a letter. The string `221B` contains only one word, namely `B`. If it were the case that `1` \rightarrow `I`, then `IB` would have to be added to a query which searched for `B`.

No examples of this type of substitution were found in the sample.

3.6 Generating All Possible Combinations

If there are s points in a term where a substitution may be made and p_i possible substitutions (in-

```
topic CF1
anyof "breast scan|mammogra"
anyof "education|training"
anyof "program"
proximity 15
near 2
anyof "quality assurance|quality control"
span key 3 1000 500 1 1
top 1000
```

Figure 1: Original distance-based query for topic CF1.

cluding no substitution) at the i th such point, then

$$k = \prod_{i=1}^s p_i - 1$$

terms must be generated, not including the original. The generation of these combinations is achieved in the pre-processor by generating the sequence of integers $1 \cdots k$. The j th integer in the sequence is interpreted as a code specifying whether or not the particular substitution occurs in the j th generated term or not. The code is unravelled using a series of modulo and integer division operations using the appropriate p_i .

3.7 Example Query Transformations

The PADRE queries used in the trial were scored using distance-based measures. [7] An example in both its original and expanded forms is shown in figures 1 and 2. Note the use of the `span` command for distance-based scoring.

4 Results

Results of the various runs are presented in table 3. The original distance-based queries located the required item 47 out of 50 times over the truth version of the data but only 44 times over the 5% degraded version. Items which were retrieved tended to fall further down the list as shown by the doubling of the average rank of the known items. The three items found in the truth version but not in the degraded version were ranked 10, 92 and 14 in the former.

```

topic CF1
anyof "bheast scan|breast scan|mammogha|mammogra|mammooha|mammoora"
anyof "education|educatlion|elucation|elucatlion|thaining|thainino|thainl
ng|thainlno|thalning|thalnino|thalnng|thalnno|training|trainino|train
lng|trainlno|tralning|tralnino|tralnng|tralnno"
anyof "phogham|phogram|phooham|phooram|progham|program|prooham|prooram"
proximity 15
near 2
anyof "quality assuhance|quality assurance|quality conthol|quality cont
rol|quallty assuhance|quallty assurance|quallty conthol|quallty control"
span key 3 1000 500 1 1
top 1000

```

Figure 2: The query for topic CF1 as augmented by the preprocessor.

Run-id	Queries	Collection	No. found	Ranked 0	Ranked < 20	Ave. rank
anu5con0	Baseline	Truth	47	14	35	84
anu5con3	Baseline	Degrade5	44	9	23	197
anu5con4	Always	Degrade5	44	10	24	187
anu5con1	Not_always	Degrade5	47	12	30	109

Figure 3: Performance of methods described in the text. The “always” queries assume the OCR always incorrectly recognises the characters which lead to substitution. The “not_always” queries assume that for each substitutable character in the input, there is a non-zero, non-unity probability that the substitution occurs. The “degrade5” collection has a 5% character error rate. The highest ranked document (returned first in the list) has rank $R = 0$. Items are only found for which $R < 1000$. Items not found are assigned $R = 1000$.

Using pre-processing to expand the queries was successful (run `anu5con1`) in restoring a considerable degree of the performance lost due to data corruption. All 47 known items were again retrieved though there was still some deterioration in their rankings. However, performance of the “always” set of queries was only slightly better than that of the baseline set.

The average query processing time for run `anu5con1` was 8.16 sec. per query compared with 6.36 sec. for the baseline run `anu5con0`. This corresponds to an increase of 28%.

5 Discussion and Conclusions

No analysis has yet been made of why the original queries failed to retrieve three of the known items.

In contrast to observations made by Taghva et al [10] and Smith and Stanfill [9], considerable deterioration in retrieval performance was observed for the queries and data used here. Three items which were located in the truth data could not be located by the original query in data with an OCR error rate of 5%.

The query augmentation methods proposed here have been shown to be quite successful in narrowing the gap between the deteriorated and the baseline retrieval performance. The size of the remaining gap may be reduced by a more careful study of the characteristic substitutions in the OCR text. The substitutions used here were derived by manual inspection of only a handful of sample pages. Less frequently occurring substitutions may easily have been missed. A more comprehensive analysis using automated comparison tools is certainly indicated.

The marked difference in performance of the “always” and “not_always” runs suggests that the model of OCR behaviour implied by the latter is more realistic (as expected).

Although the preprocessor may dramatically increase the number of terms in a query,¹ the query processing time does not increase in proportion because many of the new terms will occur infrequently or not at all. As has been previously shown [5] the time to locate all occur-

rences of infrequently occurring terms is very much shorter than that for frequent terms. Consequently, the average query processing time increased by only 28%.

It is concluded that the pre-processing algorithm described is a cost-effective method for improving retrieval performance over OCR data with a character error-rate of around 5%. Further work is needed to determine how much the performance of the method could be improved with access to a complete set of characteristic substitutions. Further work is also needed to determine whether the method is suitable for more severely degraded data, such as the TREC degrade20 set.

ACKNOWLEDGEMENT

The support of the ACSys Co-operative Research Centre and the ANU-Fujitsu CAP project is gratefully acknowledged. Thanks to NIST in the USA and various copyright holders for access to the TREC data collection.

¹By a factor of 6 in the example shown in figures 1 and 2.

References

- [1] Peter Bailey and David Hawking. A parallel architecture for query processing over a terabyte of text. Technical Report TR-CS-96-04, Department of Computer Science, The Australian National University, Canberra, <http://cs.anu.edu.au/techreports/1996/>, 1996.
- [2] Chris Buckley, Amit Singhal, Mandar Mitra, and Gerard Salton. New retrieval approaches using SMART: TREC-4. In Harman [4], pages 25–48. NIST special publication 500-236.
- [3] D.A. Grossman, D.O. Holmes, O. Frieder, M.D. Nguyen, and C.E. Kingsbury. Improving accuracy and run-time performance for TREC-4. In Harman [4], pages 433–442. NIST special publication 500-236.
- [4] D. K. Harman, editor. *Proceedings of TREC-4*, Gaithersburg MD, November 1995. NIST special publication 500-236.
- [5] David Hawking. Document retrieval performance on parallel systems. In Hamid R. Arabnia, editor, *Proceedings of the 1996 International Conference On Parallel and Distributed Processing Techniques and Applications*, pages 1354–1365, Sunnyvale, California, August 1996. CSREA, Athens GA.
- [6] David Hawking and Peter Bailey. Parallel document retrieval engine (PADRE) web page. http://cap.anu.edu.au/cap/projects/text_retrieval/, 1997.
- [7] David Hawking and Paul Thistlewaite. Relevance weighting using distance between term occurrences. Technical Report TR-CS-96-08, Department of Computer Science, The Australian National University, <http://cs.anu.edu.au/techreports/1996/index.html>, 1996.
- [8] Kwong Bor Ng and Paul B. Kantor. Two experiments on retrieval with corrupted data and clean queries in the TREC4 adhoc task environment: Data fusion and pattern scattering. In Harman [4], pages 499–508. NIST special publication 500-236.
- [9] Stephen Smith and Craig Stanfill. An analysis of the effects of data corruption on text retrieval performance. Technical Report DR90-1, Thinking Machines Corporation, Cambridge MA, 1990.
- [10] Kazem Taghva, Julie Borsack, Allen Condit, and Srinivas Erva. The effects of noisy data on text retrieval. *Journal of the American Society for Information Science*, 45(1):50–58, 1994.
- [11] Sun Wu and Udi Manber. Fast text searching allowing errors. *Communications of the ACM*, 35(10):83–91, 1992.