Text-mining based journal splitting

Xiaofan Lin
Intelligent Enterprise Technology Laboratory
HP Laboratories Palo Alto

HPL-2001-137 (R.1)
November 18th , 2002*

E-mail: xiaofan.lin@hp.com

table of contents, OCR, journal splitting, text mining, text chunking, document understanding

This paper introduces a novel journal splitting algorithm. It takes full advantage of various kinds of information such as text match, layout and page numbers. The core procedure is a highly efficient text-mining algorithm, which detects the matched phrases between the content pages and the title pages of individual articles. Experiments show that this algorithm is robust and able to split a wide range of journals, magazines and books.

Text-mining based journal splitting

Xiaofan Lin
Hewlett-Packard Laboratories
1501 Page Mill Road, MS 1126, Palo Alto, CA 94304
Email: xiaofan.lin@hp.com

Abstract

This paper introduces a novel journal splitting algorithm. It takes full advantage of various kinds of information such as text match, layout and page numbers. The core procedure is a highly efficient text-mining algorithm, which detects the matched phrases between the content pages and the title pages of individual articles. Experiments show that this algorithm is robust and able to split a wide range of journals, magazines and books.

1. Introduction

Printing-on-demand (POD) is an emerging commercial publishing field, which promises to lower the cost of short-run publishing as well as to bring new revenue streams to out-of-print publications. For publications existing in paper rather than the electronic form, the first step is re-mastering, in which the paper media is scanned and converted to some widely electronic formats such as PDF files. For periodicals people usually are more interested in specific articles rather than the whole journals or magazines, so it is desirable to have separate PDF files for individual articles. Although human operators can split a journal into separate articles, the manual processing tends to be slow and expensive. That is the motivation behind our research: Design an algorithm that enables the computers to automatically detect the title pages of journals and magazines.

In Section 2 we analyze the unique problems related to journal splitting. Then we present a text-mining based JS algorithm in Section 3. Section 4 is devoted to other related issues such as the combination of other features and the detection of content pages. Section 5 shows the experimental results and Section 6 summarizes the proposed method and gives directions for future research.

2. Problem Analysis

In this section we will define the problem more clearly in terms of what information is available. Fig. 1 is the workflow of the re-mastering process:

The first step is to scan the paper books, journals and magazines and convert them into TIFF files. Color or gray-scale images are kept to preserve all the significant information. In order to reduce file size and more importantly to enable full text search, image processing and OCR (Optical Character Recognition) follow, in which the original images are segmented into homogenous regions and stored in a compact form. OCR also generates the text information. In the last step, both the processed image and the text are embedded into the PDF files.

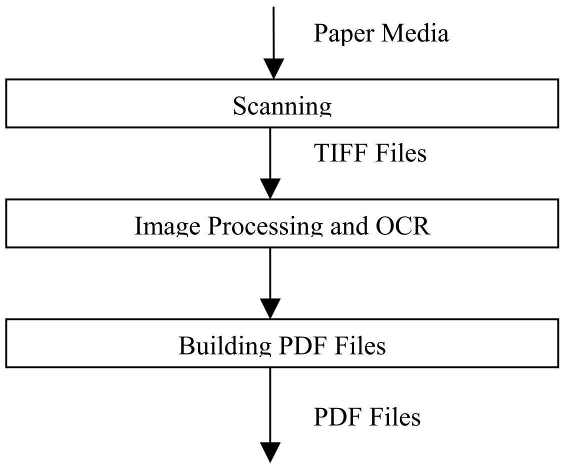


Fig 1: Workflow of the Re-mastering Process

It is apparent that two kinds of information is available for journal splitting: image data and the OCR result. Although numerous studies have been carried out on document analysis [9][10], there is no published work on the topic of journal splitting. The closest research is the logical structure analysis of books or journals by analyzing the OCR results of the table of contents (TOC) [1][2]. Similarly, one straightforward approach is to extract the page numbers from the TOC and use them to find the start page of each article [11]. There are some drawbacks with this TOC-only solution:

Scenario 1: OCR can make errors in recognizing the page numbers in the content pages.

In this situation, we get the wrong page numbers or miss some page numbers and accordingly split the journals in the wrong way.

Scenario 2: The page numbers are all correctly recognized but we will make false-negative or false-positive errors in deciding whether a digit string is a page number.

There are often other digits strings besides page numbers on the TOCs. Although we can filter out most irrelevant digit strings assuming that the distribution of page numbers on the TOC will conform to some typical patterns. For example, all the page numbers will form one or several lines. However, our research along this direction soon reveals that in real world there are so many patterns that it is an endless pursuit to include all the patterns. Worse still, it is quite common that the introduction of an extra pattern to handle one particular TOC will lead to mistakes on other TOCs.

Scenario 3: For some fancy magazines the page numbers on the TOCs and/or individual articles are printed in special formats so that the OCR engine cannot recognize them at all.

3. Proposed solution

3.1 Basic idea and problems

The study of many periodicals reveals one intrinsic characteristic common in most publications: The author names and/or the title of an article will repeat on the title page (the first page of one article). So we can detect the start pages by matching them against the content pages. On the other hand, there are several challenges along this direction:

- How to get only the "desired" matches?

We are only looking for matches on article titles and author names shown on the TOC and the title pages. In reality, the same phrases can appear in context rather than TOC or title pages. For example, they can also appear as page headers.

- How to search effectively?

Unlike the keyword-based Web search engine [6][7], we do not know in advance what phrases are the targets. The only prior knowledge is that some phrases in the TOC pages will repeat in the title pages. Without careful design, the computation complexity can easily get out of control.

- After detecting the matches, how to decide which pages are title pages?

Although it is quite safe to say that some phrases in the TOC pages will appear in the title pages, the reverse is not necessarily true. Many common words such as "the", "and", "of" will appear in almost all the pages. Even the author names can appear as citations or references instead of on title pages.

- Is text match alone sufficient for accurate decision?

In the following subsections we describe a text-mining based JS algorithm, which can overcome the first three challenges. The last challenge is addressed in the next section.

3.2 Dynamically built dictionary

A huge amount of text string matches will be involved. In order to accelerate the matches, we build a dynamic tree-structured dictionary for each content page (Fig 2). Once the dictionary is constructed, the search is conducted as nonlinear tree search instead of linear matches across the string list. Our experiment shows the speed is increased by three times in this way.

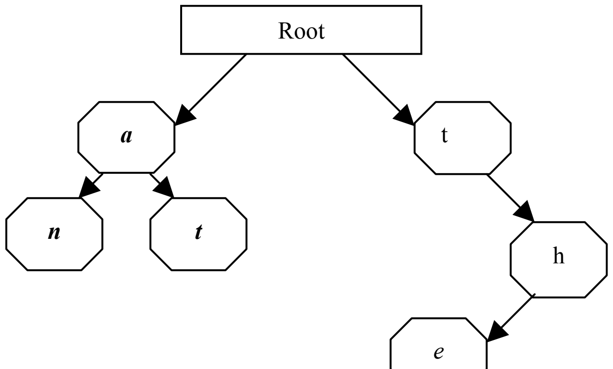


Fig 2: Tree Dictionary Structure Based on TOC Pages (Nodes that can serve as ends of words are in bold italic font.)

3.3 Graph representation of content pages

The next question is how to describe the content page. The most simple approach is to use a linked list to string together all of the words. The drawback is that repeated words will appear more than once in the list. So we model the content page as a directional graph with each node as a unique word.

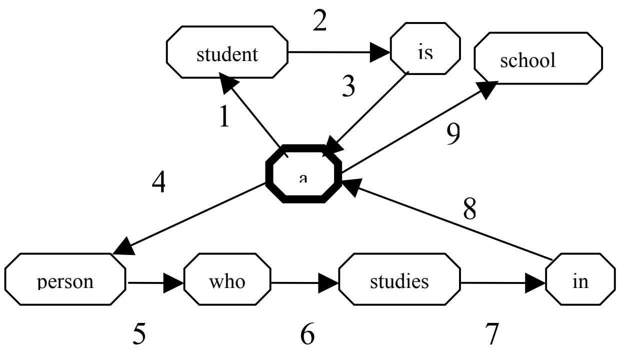


Fig 3: Graph Description of TOC Pages

Fig 3 is the representation of the sentence "a student is a person who studies in a school." Each edge has both a direction and a state and can be described as a tuple (start vertex, end vertex, state). The state component dictates the traversal order by requiring that the state be always incremented by one when traveling from one vertex to another. The biggest advantage of graph representation is the high efficiency. For example, in order to find all the phrases starting with the word “a”, we first look up the vertex in the tree dictionary. Then we can immediately find out the phrases starting from that word (“a student”, “a school” and “a person”) by traversing along all the edges from that vertex. Besides, if both the start state and the end state are given, one phrase can be uniquely decided. We define such a pair (start state, end state) as a *Range*. For example, as shown in Fig 3 Range (4,6) corresponds to the phrase “person who studies”. We use this concept in Section 3.5 when discussing text chunking.

3.4 Text match algorithm

In recent years, a lot of research is done in field of text mining, whose goal is to automatically spot important information from huge amount of text [6][7]. One recurring problem is to group documents by identifying similar phrases. In [7] the authors apply the pattern detection idea from the famous LZW data compression algorithm. In [6] a data structure called “suffix tree” is used to locate common phrases among different documents. However, these Web-related applications all utilize the statistical qualities of common phrases, such as the frequency of phrases. In journal splitting, we cannot count on the multiple appearances of critical phrases. They may only show up twice---one in the TOC and the other in the title page. So we introduce a unique text match algorithm suitable for JS. Fig 4 is the workflow of the proposed algorithm:

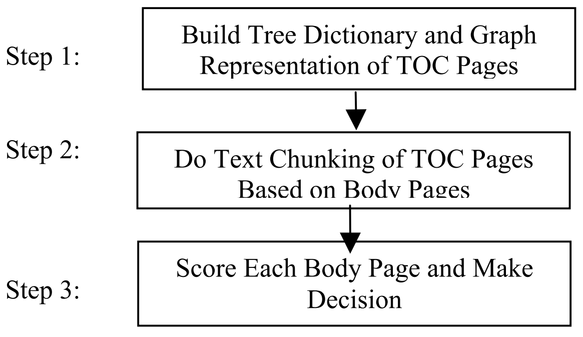

Step 1:
Build Tree Dictionary and Graph Representation of TOC Pages
Step 2:
Do Text Chunking of TOC Pages Based on Body Pages
Step 3:
Score Each Body Page and Make Decision

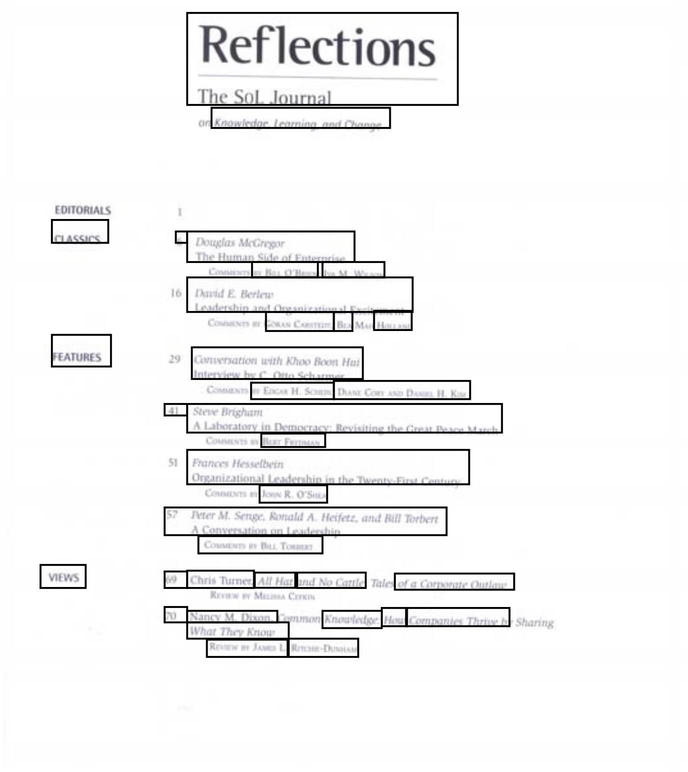
Fig 4: Major Steps of Text Mining Based JS

3.5 Text chunking

Phrases are more meaningful than individual words for the text match. So the first task is to locate key phrases in the TOC pages. We refer to this stage as “Text Chunking,” which is borrowed from natural language processing (NLP). In NLP text chunking “involves dividing sentences into nonoverlapping segments” [8] and is done in unsupervised mode by directly dividing sentences into phrases using linguistics or statistics. In the context of JS we can guide the chunking with information from body pages (non-TOC pages). Based on the heuristics that a text segment in TOC should have a corresponding maximal match in the body pages, we design the following text chunking algorithm:

```
RangeList:= {}
foreach <Word>∈ BodyPages
    Find the longest match starting with Word between BodyPages and TOC. We put this match into a Range (s,e).
    If (s,e) overlaps with any existing Range in RangeList, the two Ranges are merged. Otherwise we insert it into the RangeList.
endfor
```

Fig 5 is an example of text chunking. The words in the same rectangle are put together after text chunking. It can be seen that most article titles and author names are properly grouped together. Some phrases cannot be grouped together due to OCR errors. It should be emphasized that no geometric layout information is used in the text chunking stage. The grouping result comes exclusively from text matches.



Reflections

The SoL Journal

on Knowledge, Learning, and Change

EDITORIALS 1

CLASSICS

6 Douglas McGregor
The Human Side of Enterprise
Comments by Bill O'Brien, Iva M. Wilson

16 David E. Berlew
Leadership and Organizational Excitement
Comments by Göran Carstedt, Bea Mah Holland

FEATURES

29 Conversation with Khoo Boon Hui
Interview by C. Otto Scharmer
Comments by Edgar H. Schein, Diane Cory and Daniel H. Kim

41 Steve Brigham
A Laboratory in Democracy: Revisiting the Great Peace March
Comments by Bert Frydman

51 Frances Hesselbein
Organizational Leadership in the Twenty-First Century
Comments by John R. O'Shea

57 Peter M. Senge, Ronald A. Heifetz, and Bill Torbert
A Conversation on Leadership
Comments by Bill Torbert

VIEWS

69 Chris Turner, All Hat and No Cattle: Tales of a Corporate Outlaw
Review by Melissa Cefkin

70 Nancy M. Dixon, Common Knowledge: How Companies Thrive by Sharing What They Know
Review by James L. Ritchie-Dunham

Fig 5: Text Chunking Result for a Journal

3.6 Evaluation Phase

Although it is almost always safe to say that the title pages will be reflected by some grouped phrases in the TOC pages, the reverse is far from the truth. When there is a match between a body page and the TOC page, we cannot conclude that the body page must be a title page largely due to two reasons:

- There will be some irrelevant matches.
- Even a significant match can appear non-title page. For example, in some journals the author names or article titles will be printed across the whole article as headers/footers.

So we have to find some way to draw reliable conclusion based on the detected evidence. The following is the evaluation algorithm for that purpose:

- Assignment of each Range to only one body page.

  In the case of multiple pages mapped to the same Range, a disambiguation algorithm is applied:

```
foreach <Range>∈ RangeList
    Build a PageArray P[1,2,…,n]consisting of all the n
    pages that can be matched with Range.
    foreach <P[i]>∈ P[1,2,…,n]
        Give a match score S[i] based on a number of
        factors:
        1) Match length: the longer the overlapped
        phrase, the higher S[i] is.
        2) Page number: the first page that has a good
        match with Range should get higher score. It is to
        reduce the effect of "noise" matches such as
        those from headers.
    endfor
    Assign the score based on winner-takes-all strategy:
    1)      Find the biggest score among S[1,2…n] and
get the associated page P[m].
    2)      Credit P[m] with a score S[m] and all the
other pages with just 0.
endfor
```

- Calculation of the total score T[i] for each page P[i] by summing up all the credits from each Range.
- Select the title pages applying a given threshold.

```
TitlePageList:={}
foreach <P[i]>∈ P[1,2,…,n]
    if (T[i]>THRESHOLD)
        Insert P[i] into the TitlePageList
    endif
endfor
```

4. Other Considerations

In fact, in order to put the idea into a complete solution we still have to solve a few problems:

- Text match alone sometimes is insufficient to make accurate decision.

  It is not uncommon that some article titles consist of one or two words, or due to OCR errors only part of the title is matched. Under either condition the evidence from text match may not be enough to reach a reliable decision.

  Just as a typical pattern recognition problem, we can improve the accuracy by employing multiple classifiers or extracting more features [3][4][5]. Our strategy here is to incorporate a variety of other heuristics into the score S[i] of Section 3.6:

1) The phrases in large fonts are more likely to be titles.
2) Title phrases usually occupy the whole lines; namely, include the first and the last word of a line.
3) The match of a long word is more significant than that of a short word.

- How to decide which pages are TOC pages?

  In the above discussion, we always assume that TOC pages are known. In practice, the system should be able to automatically detect the TOC pages. For that purpose, we introduce another TOC detection phase:

1) Build a candidate set (CS) of all potential TOC pages including he first N pages and also the/back cover pages.
2) The hypothesis testing follows. For every page C[i] in CS, we first assume it is a TOC page and go through the text chunking and evaluation phase. Based on T[1,2…,n] (see Section 3.6) we use some heuristics to calculate how good C[i] is a TOC page. The idea is that a correct hypothesis should be reflected in more high scores in T. If the confidence is higher than a threshold, C[i] will be accepted as a TOC page.

5. Experimental results

We have tested the algorithms on twelve academic journals published by MIT press. Table 1 shows the result. False negative refers to errors in which the system misses a title page and false positive refers to errors in which it incorrectly adds an extra title page.

Table 1: Experimental Results

No. Title Pages	Correctly Detected	False Negative	False Positive
163	162	1	0

6. Conclusions

In this paper we analyze the unique problems around journal splitting and propose a text-mining based algorithm that combines different features to achieve robustness and high accuracy. Experimental results prove the effectiveness of this novel approach. The major characteristics of the proposed method are summarized as follows:

- **Problem Modeling**

Although it is true that JS is a brand new problem on which existing literature has almost no coverage, it can be modeled as an object classification problem with its own peculiarities:

1) Effective features used in classification are needed. Here we use text mining combined with other features.
2) It is a two-class problem: title page or non-title page.
3) The objective is to classify a sequence of objects, so the context (neighboring pages) also provides extra clues.

- **Methodology**

We employ a statistical rather than rule-based approach. In intermediate steps we avoid hard decisions by using quantitative scoring.

- **Scope**

Originally, the system is designed to handle the splitting of specific academic journals. With the text mining based approach we can easily expand the scope both horizontally and vertically. First, a wide range of generic publications such as journals, magazines and even books can be processed because the basic assumption of this method is easy to satisfy. Second, with all the matches between the TOC and body pages, we can build navigation information into the electronic documents such as PDF files so that users can be guided to individual articles when clicking on a title or author name in the TOC.

- **Future directions**

One future research topic is to go beyond linear logical structure extraction. Currently we just map all the items appearing in the content pages to body pages. Although this approach works pretty well for most periodicals, it is insufficient for some applications. For example, for books usually we are more interested in splitting individual chapters than getting all the logical components (chapters, sections, subsections and so on) without any discrimination. So the analysis of publications' hierarchy structure and the proper mapping to body pages is a topic of both research and practical value.

7. References

[1] C. Lin, Y. Niwa and S. Narita, "Logical Structure Analysis of Book Document Images Using Contents Information", Proceedings of 4th International Conference on Document Analysis and Recognition, Ulm, Germany, Aug 1997, pp. 1048-1054.

[2] S. Satoh, A. Takasu and E. Katsura, "An Automated Generation of Electronic Library based on Document Image Understanding", Proceedings of 3rd International Conference on Document Analysis and Recognition, Tokyo, Japan, Aug 1995, pp. 163-166.

[3] L. Xu, A. Krzyzak and C.Y. Suen, "Methods of Combining Multiple Classifiers and Their Applications to Handwritten Recognition", IEEE Trans. System, Man and Cybernetics, 1992, vol 22, no 3, pp. 418-435.

[4] D.S. Lee, "A Theory of Classifier Combination: The Neural Network Approach", Ph.D. Thesis, SUNY at Buffalo, Apr 1995.

[5] T.K. Ho, J. J. Hull, and S. N. Srihari, "Decision Combination in Multiple Classifier Systems", IEEE Trans. on PAMI, 1994, vol 16, no 1, pp. 66-75.

[6] P. Hannappel, R. Klapsing, G. Neumann, "MSEEC-A Multi Search Engine with Multiple Clustering", Proceedings of the '99 Information Resources Management Association International Conference, Hershey, Pennsylvania, May 1999.

[7] O. Zamir, O. Etzioni, "Grouper: A Dynamic Clustering Interface to Web Search Results", Proceedings of 8th International World Wide Web Conference, Toronto, Canada, May 1999, pp. 11-16.

[8] L. A. Ramshaw, M. P. Marcus, "Text Chunking Using Transformation-based Learning", ACL 3rd Workshop on Very Large Corpora, 1995, pp. 82-94.

[9] G. Nagy, S. Seth, and M. Viswanathan, "A Prototype Document Image Analysis System for Technical Journals", Computer, 1992, vol 25, no 7, pp. 10-22.

[10] J. Schürmann, N. Bartneck, T. Bayer, et al, "Document Analysis-From Pixels to Contents", Proceedings of IEEE, July 1992, pp. 1101-1119.

[11] A. Belaïd, "Recognition of Table of Contents for Electronic Library Consulting", International Journal on Document Analysis and Recognition, 2001, vol 4, no 1, pp. 35-45.