Intelligent Content Based Title and Author Name Extraction from Formatted Documents

Eric G. Berkowitz, Mohamed Reda Elkhadiri, Tim Sahouri, Michel Abraham

Department of Computer Science
Roosevelt University
1400 North Roosevelt Boulevard
Schaumburg, IL 60173

Abstract

This paper describes the development of algorithms for extracting the title and the names of the authors from documents available on the World Wide Web. In this paper we describe several algorithms for doing so in a manner designed not to rely on specific stylistic dictates of any document formatting standard. Rather, they are designed to rely on a combination of overt and subtle cues that form a generalized, common standard for placing this information in a document and its easy extraction by readers.

Introduction

We began work on a system designed to aid students preparing papers in the social sciences and humanities perform literature reviews. The original concept was to design an automated literature review system. We thought such a system would provide on-the-fly research assistance not equal to that of a librarian but much better than that provided by a traditional Web search (Arms 2000). The system worked as follows. A student would use a traditional search engine such as Google or Yahoo to find a paper relevant to the topic being researched. The student would then pass the document to the automated system. Using the document's citations as a form of link, (Hitchcock et. Al 2002) the automated literature review system would then attempt to perform its own search for each of the documents cited in the bibliography, thereby aiding the student in collecting a range of relevant documents. Doing so required the system to be able to perform the search, download the first few documents returned by the search, and then scan the downloaded documents to determine if one of them is indeed the document being sought. In order to determine if a given document is indeed the one being sought, the system needed to be able to determine its title and the names of the authors. Thus we began to develop our first algorithms for intelligent title and author name extraction.

Background

Human readers determine the location of the title in a document from a combination of visual and contextual cues. Font size, location, isolation, letter case and grammar all combine to provide clues to the reader that a given segment of text is the title of the document. In specific instances, such as documents formatted for publication in a specific journal, predetermined formatting characteristics dictate, from a formatting and style perspective, exactly where and in what form the title should appear. We needed to develop an algorithm to determine the author of a document in an automated way by computer. Given the resources at our disposal, we needed to develop such an algorithm without reliance on visual cues that can only be determined through graphical analysis of a fully rendered document. Thus we began to develop our first context based algorithm.

Initial Development

We specifically wanted our algorithm to work on documents written by persons working in the social sciences and humanities. One of the issues we discovered very early in our attempts to develop the algorithm is that not only are there a plethora of document formatting styles, but that these styles are rarely adhered to exactly. Thus the extraction algorithm would need to rely on heuristics that are independent of the dictates of any given style. Another issue that plagues work in this field is that there is no provably correct answer. When a human reader looks at the document he relies on a preponderance of cues to determine what he/she believes is the title. The title page or header of a document is, to the human reader, a strongly structured document part (Buamann et. al. 1995). In fact, our early experiences in reading teaches us the skills we need to do so with a high degree of accuracy, yet still, it is impossible to "prove" that a segment of text is the title and not some other segment of text, particularly with many ornately formatted documents. Thus there is no way for an automated system to validate its own results. It can not "plug the answer back into the equation" as one might do in mathematics to verify a determination.

In order to avoid parsing a document formatting language, the first algorithm we developed relies on the program pdftotext that comes with the xpdf package to rerender a document in PDF format as a text document while attempting to preserve, as much as is possible, the original layout of the document.

The initial algorithm we developed assumes that the title of a document must come at or near the beginning of the text and thus is scans the first 100 lines of text in a document. It then scan these lines to find the title of the document and the names of the authors using a set of positive and negative heuristics. Positive heuristics are those that increase the likelihood that a line of text is part of the elements being searched for, that is, part of the authors' names or title. Negative heuristics are those that decrease the likelihood that a line of text is part of an element being searched for.

The algorithm first attempts to determine if the text being looked at is part of another element commonly found either before the title and authors' names in a document or in close proximity to them. A sample of these elements and the patterns used to form Java syntax regular expressions that constitute the heuristic test performed on a line of text to qualify it as probably being that element is shown in Table 1. The tests for an association assumes the author is associated with a school or institute. A line of text that matches one of these negative heuristics is determined not to be part of the title or authors' names. These lines are discarded by the algorithm.

Element	Pattern
e-mail address	"@"
Association	" [Uu]niversity of" "University" "Department" "Dept" "College\\s*\\z" "Institute"
City, State	"\\b[A-Z][A-Za-z]+,\\s+([A-Z]{2,})\\b" "\\b[A-Z][A-Za-z]+,\\s+([A-Z][A-Za-z]+)\\b"
URL	"http"
Phone number	"\\d{3,3}[\\s]*-[\\s]*\\d{3,}"
Volume Reference	"(\\b[Vv]ol[\\.:]\\s*\\d+)|(\\bp\\.\\s*\\d+)|(\\bpp.\\s*\\d+)|(\\b[Nn]o\\s*\\d+)"
End of Header	"Abstract"
Body Text	"\\b([a-z\\-]+[,\\.!\\?]*\\s+){4,}"
Copyright	"\\b[aA]ll [rR]ights [rR]eserved\\.*\\b"
Credits	"are a" "is a" "is the"
Date	"\\A\\s*\\w+\\s+\\d{4,4}\\s*\\Z"
Citation	"[Pp]resented\\sat\\sthe\\s.+?[Cc]onference" "\\b\\s*Journal of"
Blank	"^\\s*$"
Other Element	"[%\\|\\[{=\\$#!<>\\?]"
Too Short	"\\S+(\\s+\\S){2,}"
Prepostion or ucommon title word	" of ", " the ", " at ", " for ", " in "
Four+ lower case words	"\\b([a-z\\-]+[,\\.!?]*\\s+){4,}"

Table 1

The negative heuristics are followed by a set of positive heuristics that attempt to determine whether the text is part of the authors' names or part of the title.

Initial Results and Enhancements

Comment	Pattern or other Test
List of names	"\\S\\s+,\\S"
Title already found must be names	Title != null
Two digit numbers can only appear in the title	"\\d{2,}"

Table 2

The initial results of our algorithm were very promising and we were successful in extracting appropriate text from a wide variety of documents in a surprisingly large fraction of the test cases. The algorithm was, however far less successful at distinguishing between the title and the authors' names. While it was quite successful at excluding most parts of an address, it was including the city/state/country part of an address in a large number of cases, particularly in non-U.S. addresses where the format of <city>, XX where XX is a two letter state abbreviation, does not apply. Even with U.S addresses the algorithm was failing to remove the city/state part of an address if the author spelled out the state name instead of abbreviating it.

The algorithm was therefore enhanced with the ability to directly recognize many proper names and the names of all U.S states, Canadian provinces, and foreign countries. A list of most common surnames was retrieved from the U.S Census Bureau and a list of states and countries was created using data from the USPS and other sources on the Web.

The test for and address line that searched for data of the form <city>, XX was replace with a two step algorithm that test first for data of the form <city>, <state> where <city> and <state> can be of any length. If such text is found the second string, <state> is tested against the lists of countries, states, and abbreviations. If it is found in the list, this line of text is treated as part of an address and discarded. This significantly reduced the amount of address information slipping through the tests.

Distinguishing between names and title text was the next difficulty we tried to resolve. The positive heuristics already incorporated in the algorithm that were shown to be inadequate for making this determination were used as

the foundation of a scoring system. Each line of text that passed the earlier tests and was therefore determined to be either part of the title or the authors' names is passed through a sequence of tests. Two scores are maintained: one indicating the likelihood it is part of the title and the other indicating the likelihood it is the authors' names. The first scoring tests are the heuristics shown earlier. Next each line of text is run through a spell checker. The ratio of misspellings to the total number of words is calculated. If more than 80% of the words in the line are determined not to be correctly spelled English words, the score for being more likely names is incremented. While this test particularly stands out as limiting the algorithm to papers written in English, a review of the previous heuristics will show that, they too, assume English text and so using only an English dictionary does not reduce the scope of the algorithm's applicability. The next scoring test is to check whether the line starts with a surname from the list retrieved from the Census Bureau. If it does the names score is incremented again. Next the system checks if any previous lines were determined to be part of the title. If not, the system assumes that the title will precede the authors' names in the paper and increases the title score. If however, some text has been determined to be part of the title and there has been at least one blank line encountered since that text was found, the name score is incremented. A check for punctuation is performed. If commas are present in the line, given that addresses are presumed to already have been excluded, the name score is incremented. The final scores are then used to determine the nature of the text being checked. This algorithm is now named algorithm N-1 and the success of this algorithm will be shown in the section comparing algorithm performance at the end of this paper.

A Format Based Approach

Algorithm N-1 uses a long list of heuristics that rely on content alone to make determinations about the text. As a next stage in development we decided to take a format based approach. In a large number of document styles, the title is required to have the largest font on the first page of the document. Even in documents that do not adhere specifically to any given style, it has become a ***de facto*** standard to display the title in larger font than the rest of the document. One significant exception to this is the title page of a thesis or dissertation; yet still we decided to test the success of such an approach.

We began using components from the free PDFbox project, a freely available PDF parsing engine written entirely in Java. Instead of reducing the document to text we started working directly with the PDF formatted documents. First we developed a system to produce an XML formatted description of text segments and font information from a PDF, filtering out the rest of the PDF information in which we were not interested. Figure 1 is an excerpt of XML from analyzing a PDF document.

```
<?xml version="1.0"?>
<document
name="/home/eric/crawler/docdirs/a/NoT
itles/www.northwestern.edu_ipr_publica
tions_papers-mcc.pdf"><page
number="1">
<text fontName="ZapfDingbats"
fontSize="18.0">t</text><newline/>
<text fontName="Courier"
fontSize="24.0">Program on Community
Development</text><newline/>
<text fontName="Courier-Bold"
fontSize="30.0">Mapping
Community</text><newline/>
<text fontName="Courier-Bold"
fontSize="30.0">Capacity</text><newlin
e/>
<text fontName="Courier"
fontSize="13.0">NORTHWESTERN
UNIVERSITY<newline/>
...
</page>
<page number="2">
<text fontName="Helvetica-Bold"
fontSize="18.0">Mapping Community
Capacity</text><newline/>
<text fontName="Helvetica"
fontSize="12.0">by</text><newline/>
...
</page>
</document>
```

Figure 1

The next step was to create a parser for the XML data that would return one or more segments of text based on a set of criteria passed in. Our initial attempts to use this system were not highly successful. We determined that PDF documents incorporate elements other than text that are stored internally as text such as embedded graphics. In algorithm N-1 these elements were removed when the document was rerendered as text so we had not encountered them before. In order to avoid returning the textual gibberish that we were getting from searching for the largest font, we developed a two part test. The test searched for the largest font used on a "reasonable" segment of text where a reasonable segment of text is one that contains at least two three-letter words ("\\w{3,}\\s+\\w{3,}"). The test determines this font and then returns all of the text that uses the same[1] font size on the first page. This enhancement yielded significantly improved results. This algorithm however, has no mechanism for

[1] The PDF format uses floating point font sizes so "same" is within a margin of tolerance.

finding the names of the authors since we could not determine any generalized formatting convention for them. Thus it is only applicable for finding the title of a document. This algorithm appears as algorithm R-1 in the comparison at the end of this paper.

An Improved Content Based Approach

Given both approaches described above, the obvious best approach would be to combine their title extraction abilities and therefore increase the probability that we would find the title of the document. Unfortunately, the fact that given a segment of text one believes is the title of a document, there is no way to prove its correctness left us with no real way to perform this union. The algorithms use completely different approaches to finding the title and in the case where one returns information and the other does not, it is simple to use the only information available. But what of the more common case where information is returned by both and the information returned is not the same but overlaps or completely differs? How can one result be demonstrated programmatically to be superior to the other? It was therefore decided to perform a manual study of cases where algorithm R-1 performed better that algorithm N-1 and directly engineer into algorithm N-1 heuristics and mechanisms to handle these cases along with its current heuristic methods yield a single result. The result of this work was algorithm N-2.

The first lesson we learned was that while N-1 would occasionally truncate a title, R-1 would usually not truncate the same title. Analysis revealed that this occurred in the case where the title extended onto a second or third line and the last fragment was on one or two words long. R-1 would find the text since it would use the same font as the rest of the title but N-1 would disqualify the fragment as being too short to be a title. N-2 therefore incorporates a new heuristic that requires the first line of a title to have more than two words but allows subsequent lines to be shorter.

We also learned through our work on R-1 that PDF documents contain meta-information about the document that may include a title element. If there is a meta-title in the document it may or may not be the actual title of the document. A review of several common tools used to produce PDF documents revealed an assortment of behaviors regarding the generation of a meta-title. All tools allow an author to specify a meta-title when producing the PDF. If the author does do so the meta-title will probably be the actual title of the document. The manner in which the tools handle the case where no meta-title is specified differs greatly. Some tools use the filename of the PDF file being generated as the meta-title. Others use the name assigned by the operating system to the job when the PDF producing code is running. Others retain the title from document to document so that if an author once specifies the meta-title for a document that title will persist as the title of all subsequently produced documents until it is changed. N-2 utilizes this meta information in the following way. It first determines if the document contains a meta-title. If it does it searches to determine whether the meta-title found actually appears on the first page of the document. If it does, it is assumed that this is the title of the document. If it does not appear on the first page it is ignored. In the case where the meta-title is used, N-2 still attempts to find the title in the document as part of the process of searching for the authors' names, however the title found using the heuristic method is discarded and only the authors' names are used. N-2's use of the meta-title is shown in Figure 2.

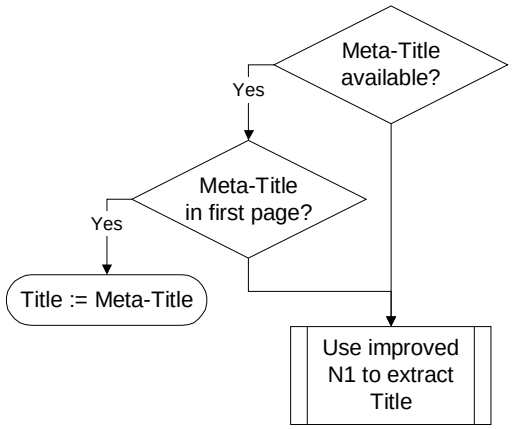

Figure 2

In reviewing the cases that N-1 missed the title and R-1 did not, we noticed that there is a large number of papers in which the title is not capitalized. R-1 was able to find the title via the formatting information while N-1 discarded the title assuming that a sequence of four or more lower case word signifies a regular sentence and not a title or list of authors' names. N-2 therefore does not automatically discard such sequences. Instead it only discards them if text from the title has been found and is detected to be set in initial-caps or if at least two blank lines have been detected since the ending text of a detected title. While this is not a perfect filter and lets some text through that it should not, it no longer excludes text that should be included as the earlier heuristic did. The full operation of N-2 is shown in Figure 3.

Performance Comparison

The following tables presents the relative performance of the algorithms on a random sample of 100 papers.

N-1	Exact Title	50.00%
	Title + Extra Text	11.00%

	Exact Title	50.00%
	Partial Title	13.00%
	Title+Author as Title	7.00%
	Missed Title	20.00%

R-1	Exact Title	53.00%
	Title+Extra Text	24.00%
	Partial Title	3.00%
	Title Missed	20.00%

N-2	Exact Title	43.00%
	Title+Extra Text	11.00%
	Partial Title	8.00%
	Title+Author as Title	5.00%
	Title Missed	32.00%

Intersection of Performance of All Algorithms	Exact Title	17.40%
	Title Missed	4.00%

Analysis

From the data in the performance table we can see that the changes to N-1 that yielded N-2 had an interesting effect. While reducing the overall chance of extracting the title text from a document, they also significantly increased the chances that if a title is returned, it will be the correct one. Accuracy was increased while overall effectiveness was decreased. Neither of the two heuristic algorithms has the effectiveness or accuracy of the formatting based algorithm, R-1.

What is perhaps more interesting is the intersection data. Only in 17.4% of the cases did all the algorithms extract the title correctly. Only on 4% of the cases did all three algorithms completely miss the title. An algorithm combining N-1, N-2 and R-1 should be able to achieve 96% efficiency although at varying degrees of accuracy.

Conclusion

The development effort that has gone into the creation of the algorithms described here has yielded interesting information both on the nature of formatted documents

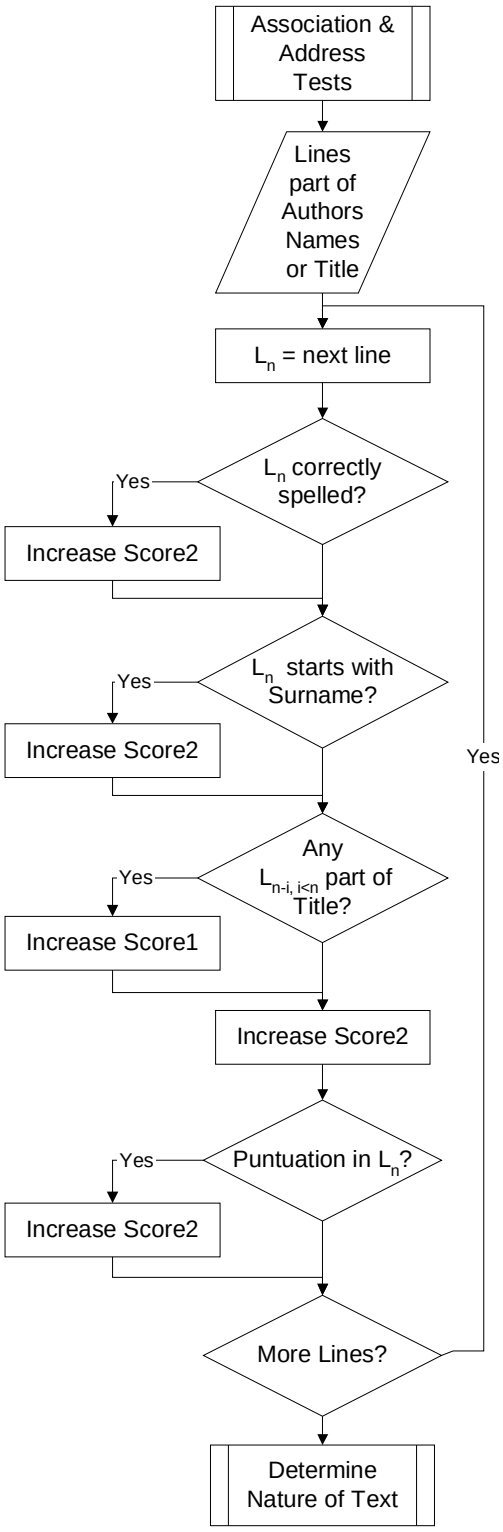

Figure 3

and appropriate methods for extracting information from them. The analysis shows that a hybrid approach should yield an algorithm capable of determining the title of a vast majority of documents without constraints on style and formatting. The major impediment to combining N-1, N-2 and R-1 is, as described earlier, the fact that we have no automated manner in which to verify correctness of title information. Thus, while human analysis can determine that, for example, in one case the title found by R-1 is correct while the title found by N-1 is partially correct, we have no method to make this determination programmatically, on the fly. Thus we are currently attempting to create a unified algorithm that combines elements of N-1, N-2 and R-1 but yields a single result.

There also remains a third approach to this analysis. While we have done much work on analyzing documents based on content and formatting, we have only begun to investigate analyzing them graphically using automated visual tools. Such analysis should yield a system that more closely mimics the activities of a human reader and should provide more insight into how to find desired information in documents.

References

William Y. Arms, "Automated Digital Libraries: How Effectively Can Computers Be Used for the Skilled Tasks of Professional Librarianship" D-Lib Magazine 6 (7/8) July/August 2000

Stephan Baumann et. al. "Document Analysis at DFKI: Part2: Information Extraction" Deutsches Forschungszentrum für Künstliche Intelligenz, RR-95-03, 1995.

Steve Hitchcock et. Al, "Open Citation Linking: The Way Forward" D-Lib Magazine, 8 (10) October 2002.