Automated Document Labeling for Web-based Online Medical Journals

Daniel X. Le and George R. Thoma
Lister Hill National Center for Biomedical Communications
National Library of Medicine
Bethesda, MD 20894

ABSTRACT

An increasing number of publishers are using the Internet and the World Wide Web to provide their subscribers with access to online journals. New techniques are needed to capture, classify, analyze, extract, modify, and reformat Web-based document information for computer storage, access, and processing. An R&D division of the National Library of Medicine (NLM) is developing an automated system, temporarily code-named WebMARS for Web-based Medical Article Records System, to download, analyze and extract bibliographic information from Web-based journal articles to produce citation records for its MEDLINE® database. This paper describes one component of this system: assigning meaningful labels to text zones containing article title, author names, affiliation, and abstract. This labeling technique is based on features derived from the World Wide Web Consortium Document Object Model (W3C DOM) and an analysis of the page layout for each journal, a DOM-based document node location and content analysis, string pattern matching, and a depth-first node traversal algorithm.

Experiments carried out on a variety of Web-based medical journals have proved the feasibility of this automated document labeling approach. Preliminary evaluation results on a small set of Web-based medical journal articles show that the system is capable of labeling text zones at an accuracy of over 95%.

Keywords: W3C Document Object Model, Automated document labeling, MEDLINE® database, National Library of Medicine.

1. INTRODUCTION AND BACKGROUND

Today, with the rapid advance and popularity of the Internet and the World Wide Web technologies, an increasing number of journal publishers provide their subscribers with access to online journals. However, Web-based online journals pose new challenges in the areas of automated document analysis and content extraction, database citation records creation, and other document-related applications. The Lister Hill National Center for Biomedical Communications, a research and development division of the National Library of Medicine (NLM), is conducting research in the area of Web-based document analysis including Web page downloading and classification, and Web-based document content extraction, modification, labeling and reformatting. To take advantage of journal information available over the Internet, we are developing an automated system code-named WebMARS to create citation records for the NLM's MEDLINE® database from online journals. This system (1) downloads and classifies Web document articles as abstract, full text, PDF or image files, (2) converts PDF files to HTML files, if necessary, (3) parses HTML files to create text zones, and labels these text zones, (4) extracts, modifies and reformats the citation information in labeled text zones to be reconciled (validated) by an operator, and finally (5) uploads the citation records to another NLM database for indexing by experts. This paper describes the automated labeling of text zones in Web-based document articles as titles, authors, affiliations, and abstracts

Most document labeling techniques proposed so far in the literature deal with scanned document images, while Web-based document labeling algorithms classify the entire Web page. In this paper, we propose an automated technique to label Web-based text zones using the W3C DOM and an analysis of the page layout for each journal, a DOM-based document node location and content analysis, string pattern matching, and a depth-first node traversal algorithm. Preliminary

evaluation results show that the system is capable of labeling online medical journals with very high accuracy. In addition, we conduct research to automate labeling of bibliographic data extracted from medical online journals using statistics and a fuzzy rule-based algorithm [1].

The rest of this paper is divided into six sections. Section 2 provides a system overview. Section 3 presents the DOM features and algorithms. Section 4 describes the Web-based document labeling process. Experimental results and summary are in Sections 5 and 6.

2. SYSTEM OVERVIEW

The automated labeling technique described here is one component of our WebMARS prototype system under development [2]. The labeling process takes a downloaded Web-based document article as its input, creates its DOM tree structure, normalizes the tree structure by eliminating unnecessary tags, builds the document string patterns and performs the string pattern matching algorithm to label each text zone as title, author, affiliation, abstract, or unidentified.

The calculated features include both position-based and content-based ones [3]. The position-based features refer to the locations of nodes in a DOM document structure such as parent and child nodes. In addition to the parent and child relationship feature, the node levels in the DOM tree structure are also considered in our work and they help to further improve the confidence level of the labeling results. The content-based features deal with the tag and/or text contents and their attributes such as font attributes or tag names.

3. DOCUMENT OBJECT MODEL FEATURES AND ALGORITHMS

"The Document Object Model (DOM) is an application programming interface (API) for HTML and XML documents. It defines the logical structure of documents and the way a document is accessed and manipulated."[4]. Figure 1 shows HTML code for a typical medical article, and Figure 3 presents a corresponding DOM-based document tree diagram.

The DOM structure consists of nodes organized as a tree; there are two kinds of nodes: tag element, and non-tag element. The difference between these two kinds of nodes is that the tag element node has a *tagName* property that is the name of the tag it represents. For example, in Figure 3, the <H2> element has a *tagName* of H2 while the node ("Title") located under <H2> node is a non-tag element node. Since documents in the DOM have a logical tree-like structure, there are two node traversal algorithms available: depth-first and breadth-first. In this paper, we use the depth-first node traversal algorithm to visit each node of the DOM tree structure.

Each node is assigned a level number based on its position in the DOM tree structure. Starting from the root node which is the <HTML> tag element node; its node level is assigned as 1. In this paper, both tag-element and non-tag element nodes as well as their node levels are used to generate string pattern features for the labeling process. Furthermore, the labeling algorithm deals with text zones only; therefore in order to reduce the number of string patterns generated, several unnecessary non-text tag elements such as <FORM>, <FRAME>, <IMG>, <TD>, and <TR> are eliminated in the normalization process. These nodes are marked as "not-used", while other nodes are marked as "available".

Features calculated for this labeling technique are based on an analysis of the page layout for each journal and the layout's associated DOM structure. Two types of DOM-based string patterns are considered in this paper: level-pattern and tag-pattern. Each level and tag is separated by dots (".").

Using this normalized DOM structure and starting from the root node, the system uses the depth-first node traversal algorithm to visit "available" nodes and builds the level-pattern and tag-pattern strings for each document article. Figure 2 presents an example of both level-pattern and tag-pattern strings for the HTML-based article shown in Figure 1. These strings are then compared against predefined Web article string patterns to label text zones with labels such as titles, authors, affiliations, and abstracts.

4. WEB-BASED DOCUMENT LABELING PROCESS

The labeling process consists of four steps: (1) download a document article from a publisher

Web site and create its DOM structure, (2) normalize the DOM structure, (3) build document article string patterns, and (4) perform string pattern template matching against Web document articles that follow a predefined layout structure to associate a label with each zone of interest. In the following subsection, each step is discussed.

4.1 Download a document article and create its DOM structure

In this step, the system connects to a publisher Web site, selects a particular journal issue, and downloads its document articles. Note that in a production system this step must be permitted by copyright and subscription agreements. The successfully downloaded articles are stored as HTML files in a directory in the file server. The downloading process is repeated until all document articles of the selected journal issue are downloaded. Following the downloading step, a DOM structure is created for each document article.

4.2 Normalize the DOM structure

In this step, unnecessary non-text tag elements such as <FORM>, <FRAME>, <IMG>, <TD>, and <TR> are eliminated to shorten the length of generated string patterns and to improve the speed performance of the labeling algorithm. Normal nodes are labeled as "available" while eliminated nodes are labeled as "not-used".

The following algorithm summarizes the normalization process to reorganize the DOM structure of a document article being labeled.

1. Initialize all tag and non-tag elements as "in-process".
2. Mark all heading tag elements <H1> to <H6> and all of their non-tag element children as "available"
3. Eliminate all "in-process" children elements under <TABLE> tag.
4. Mark all non-tag elements as "available"
5. Eliminate all "in-process" elements having tag names as follows: <!>, <A>, <ACRONYM>, <ADDRESS>, <APPLET>, <AREA>, <BASE>, <BASEFONT>, <BDO>, <BGSOUND>, <BLOCKQUOTE>, <BUTTON>, <CAPTION>, <CODE>, <COL>, <COLGROUP>, <COMMENT>, <CUSTOM>, <DD>, <DEL>, <DFN>, <DIR>, <DIV>, <DL>, <DT>, <EMBED>, <FIELDSET>, <FORM>, <FRAME>, <FRAMESET>, <IFRAME>, <IMG>, <INPUT>, <INS>, <ISINDEX>, <LABEL>, <LEGEND>, <LI>, <LINK>, <MAP>, <MARQUEE>, <MENU>, <META>, <NOFRAMES>, <NOSCRIPT>, <OBJECT>, <OL>, <OPTGROUP>, <OPTION>, <PARAM>, <PLAINTEXT>, <Q>, <RT>, <RUBY>, <SAMP>, <SCRIPT>, <SELECT>, <SPAN>, <STYLE>, <TBODY>, <TD>, <TEXTAREA>, <TFOOT>, <TH>, <THEAD>, <TR>, <UL>, <VAR>, <XML>, <XMP>.
6. Filter out any non-tag elements having empty node value.
7. With the exception of <HTML>, <HEAD>, <TITLE>, and <BODY> tag elements, eliminate any "in-process" elements that do not have non-tag element children.

4.3 Build document article string patterns

Using the normalized DOM structure from the previous step and the depth-first node traversal algorithm, the system builds two string patterns for each document article: the level-pattern string and the tag-pattern string. For each text zone, the system records node level values as well as tag names till a non-tag element is encountered. The numbers in each level-pattern string are separated by a dot (".") and the tags in each tag-pattern string are also separated by a dot (".). Figure 2 presents an example of both the level-pattern and tag-pattern strings of the HTML-based article shown in Figure 1.

4.4 Perform string pattern template matching

Finally, the string pattern matching algorithm compares the two kinds of string patterns generated in the above step against a predefined string pattern of Web articles of similar journal issues. Using both types of string patterns allows the labeling algorithm to assign a level of confidence in each labeled text zone. The confidence information eventually will be analyzed to further improve the matching algorithm as well as to provide some insights into the features selected for each document article.

5. EXPERIMENTAL RESULTS

The labeling technique described above has been implemented, and experiments have been conducted with Web-based document articles selected from several different medical journals. All documents used in these experiments have

HTML-based file format. Samples consisting of 855 document articles covering a wide variety of layouts were downloaded from 6 different journal publisher websites to create training and testing data sets. The training data set consisting of 229 document articles was used to create six sets of string patterns for the predefined document layout structures of six journal issues. The test data set, consisting of the remaining 626 document articles, is used to estimate the labeling accuracy. Preliminary results show that the average labeling accuracy on the test data set was over 95.0 %. Most errors were due to the incomplete collection of predefined document layout string patterns of journal issues.

6. SUMMARY

We have presented a technique for the automated document labeling of Web-based online medical journals using the W3C Document Object Model and string pattern template matching. The technique yielded very good performance on a set of 855 document articles drawn from 6 different medical journals, and showed the feasibility to label HTML-based document articles effectively, and the possibility of extension to other medical journals. Moreover, in this work, the font sizes and font attributes (normal, bold, underlined, italics, and fixed pitch) were not used in the labeling process. Since text zones in a document article usually employ different font sizes and/or font attributes for differentiation, we plan to expand the current labeling features to include the font information in order to further refine the automated document labeling for Web-based online medical journals.

7. REFERENCES

[1] J. Kim, D. X. Le, and G. R. Thoma, "Automated Labeling of Bibliographic Data Extracted From Biomedical Online Journals," *Proc. SPIE,* Document Recognition and Retrieval X, Santa Clara, CA, Vol. 5010, pp. 47-56, 2003.

[2] D. X. Le, L. Q. Tran, J. Chow, J. Kim, S. E. Hauser, C. W. Moon, and G. Thoma, "Automated Medical Citation Records Creation for Web-Based Online Journals," *The Fourteenth IEEE Symposium on Computer-Based Medical Systems*, Bethesda, MD, pp. 315-320, 2001.

[3] J. Marini, The Document Object Model: Processing Structured Documents, McGraw-Hill/Osborne, California (2002).

[4] http://www.w3c/DOM


```
<HTML>
<HEAD>
<TITLE>This is a document</TITLE>
</HEAD>
<BODY>
<H2>Title</H2>
<P><STRONG><NOBR>Author</NOBR></STRONG></P>
<P><FONT SIZE=-1>Affiliation</FONT></P>
<P>Abstract.</P>
<P>< STRONG >Key Words</ STRONG ></P>
</BODY>
</HTML>
```

Figure 1: An example of the HTML code for an online medical article.

Level-patterns	Tag-patterns	Node contents
1	HTML	
2	HEAD	
3	TITLE	This is a document
2	BODY	
3	H2	
4		Title
3.4.5	P.STRONG.NOBR	
6		Author
3.4	P.FONT[,,-1]	
5		Affiliation
3	P	
4		Abstract
3.4	P.STRONG	
5		Key Words

Figure 2: An example of DOM-based string patterns

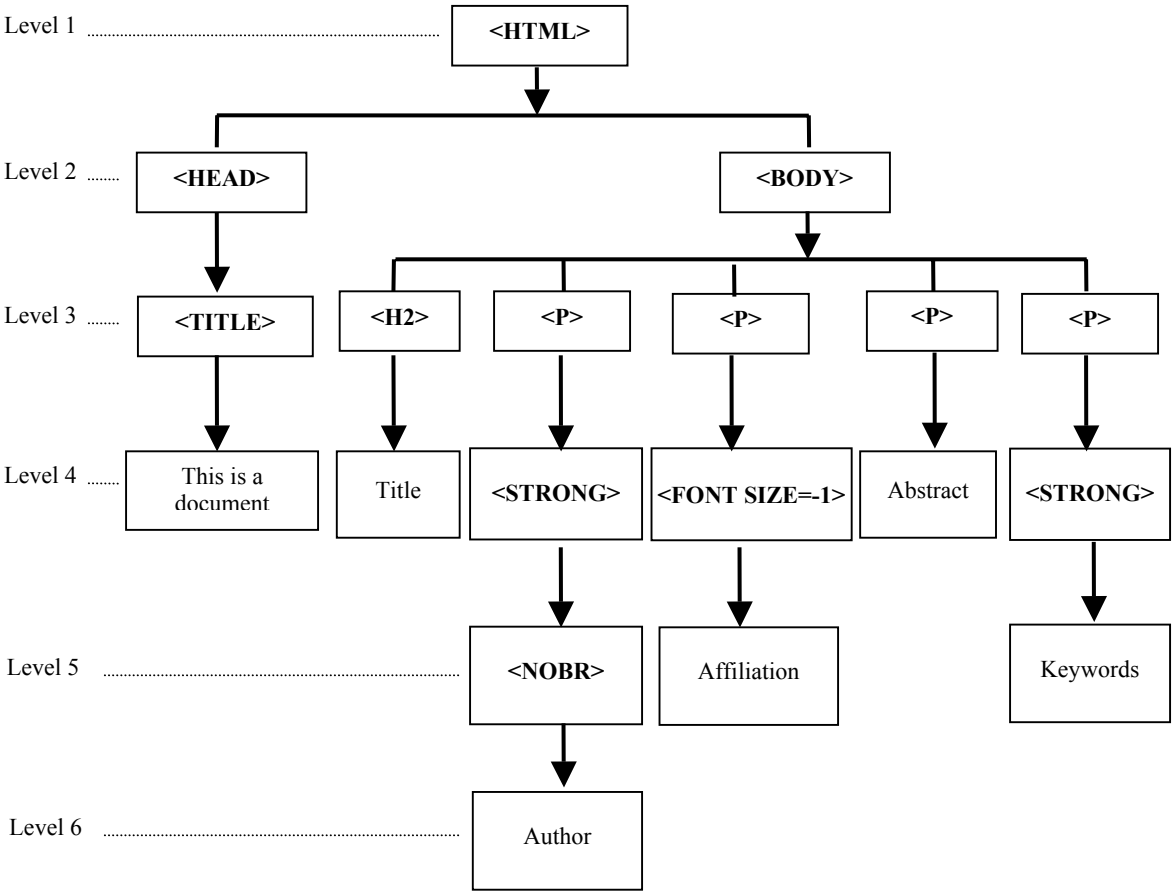

Figure 3: A DOM-based document tree diagram