Web Document Text and Images Extraction using DOM Analysis and Natural Language Processing

Parag Mulendra Joshi, Sam Liu

HP Laboratories
HPL-2009-187

Keyword(s):
Web page text extraction, Image extraction, Natural language processing, HTML documents, DOM trees

Abstract:

Web has emerged as the most important source of information in the world. This has resulted in need for automated software components to analyze web pages and harvest useful information from them. However, in typical web pages the informative content is surrounded by a very high degree of noise in the form of advertisements, navigation bars, links to other content, etc. Often the noisy content is interspersed with the main content leaving no clean boundaries between them. This noisy content makes the problem of information harvesting from web pages much harder. Therefore, it is essential to be able to identify main content of a web page and automatically isolate it from noisy content for any further analysis. Most existing approaches rely on prior knowledge of website specific templates and hand-crafted rules specific to websites for extraction of relevant content. We propose a generic approach that does not require prior knowledge of website templates. While HTML DOM analysis and visual layout analysis approaches have sometimes been used, we believe that for higher accuracy in content extraction, the analyzing software needs to mimic a human user and under- stand content in natural language similar to the way humans intuitively do in order to eliminate noisy content. In this paper, we describe a combination of HTML DOM analysis and Natural Language Processing (NLP) techniques for automated extractions of main article with associated images from web pages.

External Posting Date: August 21, 2009 [Fulltext] Approved for External Publication
Internal Posting Date: August 21, 2009 [Fulltext]

To be published in the 9th ACM Symposium on Document Engineering, DocEng'09, Munich, Germany. September 16-18, 2009

Web Document Text and Images Extraction using DOM Analysis and Natural Language Processing

Parag Mulendra Joshi
Hewlett-Packard Laboratories
1501 Page Mill Rd
Palo Alto, CA 94304, USA
parag.joshi@hp.com

Sam Liu
Hewlett-Packard Laboratories
1501 Page Mill Rd
Palo Alto, CA 94304, USA
sam.liu@hp.com

ABSTRACT

Web has emerged as the most important source of information in the world. This has resulted in need for automated software components to analyze web pages and harvest useful information from them. However, in typical web pages the informative content is surrounded by a very high degree of noise in the form of advertisements, navigation bars, links to other content, etc. Often the noisy content is interspersed with the main content leaving no clean boundaries between them. This noisy content makes the problem of information harvesting from web pages much harder. Therefore, it is essential to be able to identify main content of a web page and automatically isolate it from noisy content for any further analysis. Most existing approaches rely on prior knowledge of website specific templates and hand-crafted rules specific to websites for extraction of relevant content. We propose a generic approach that does not require prior knowledge of website templates. While HTML DOM analysis and visual layout analysis approaches have sometimes been used, we believe that for higher accuracy in content extraction, the analyzing software needs to mimic a human user and understand content in natural language similar to the way humans intuitively do in order to eliminate noisy content.

In this paper, we describe a combination of HTML DOM analysis and Natural Language Processing (NLP) techniques for automated extractions of main article with associated images from web pages.

Categories and Subject Descriptors

H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing—*Linguistic Processing*; H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Information Filtering*

General Terms

Algorithms, Experimentation, Verification

To appear in the ACM Document Engineering Conference, September 2009 at Munich, Germany

Keywords

Web page text extraction, Image extraction, Natural language processing, HTML documents, DOM trees

1. INTRODUCTION

Considering that a huge amount of world's information resides in web pages, it is becoming increasingly important to analyze and mine information from web pages. Applications vary from Information Retrieval (IR) to creation of more appealing and device specific content re-use, such as creating alternate versions for printing or for viewing on small PDA screens. However, HTML format is designed more for human users to view and does not easily lend itself to automated processing for information extraction. A seemingly simple visual block containing a few paragraphs of text is typically coded using tens of HTML nodes some of which contain the text and others contain styling information to control layout. This problem is compounded by the fact that typical web pages contain significant amount of unrelated content mixed in with the main content. The noisy content is typically in the form of navigation bars on top and/or on the side, horizontal or vertical banner ads, boxes with links to related content, boxes containing images or animated ads, etc. Such noisy content adversely affects performance of web content analysis and information extraction technologies.

Therefore, it is essential to extract the main article content including all the text with original styling information along with any associated images or figures with their captions while eliminating other images which are advertisements and other unrelated text blocks.

In this paper, we first describe related work and applications. Then we give a detailed problem statement and finally we describe our approaches to solve the problem.

2. BACKGROUND AND RELATED WORK

In the paper by Laender et al[5], many existing approaches are described. These approaches rely on writing wrappers for a type of web pages. Some of these approaches rely on hand-crafted web scrapers that use hand-coded rules generally specific to known template types. Since some large websites generally use a set of templates, these templates are studied and website-specific rules are created that look for certain patterns in HTML nodes or text. Rules are typically written in traditional programming language or specialized tools like NoDoSE[1] and XWRAP[7].

Problem with such approaches is that they are very brittle

and break when websites change their templates or layout styles. So, they need to be constantly maintained. Also, this approach is not scalable as it requires logic to be hand-coded by programmers for each website or each template style.

Methods described in[6],[10] and[8] try to infer a template by analyzing several web pages created using the same template and the limitations of such approaches are that they require collection of a large set of web pages for each template style and secondly, they assume that the noisy elements would be repeated across pages in the same places.

Methods such as, VIPS[2] rely on visual layout information to isolate main content from the noisy content. By looking at rendered web page, the VIPS algorithm tries to perform visual segmentation. The main limitation of this approach is that performing visual rendering and segmentation of web pages is resource intensive.

Method[9] uses a mostly machine learning approach for extraction of text. Method[4] uses DOM analysis heuristics some of which rely on a list of advertisement servers. Both of these methods only extract the article text. Our approach extracts not only the text but also associated images based on semantic similarity of image captions to the main text.

Collective limitation of all of these previous cited approaches is that they do not perform any semantic analysis. A human being relies on a number of cues and the most important of them is reliance on semantic similarity. For example, a person quickly looks at a news article and determines which person / organization it is about and tries to decide relevance of all other content blocks based on that understanding.

In the current work, we have taken steps along that approach.

3. APPLICATIONS

Intelligent isolation of main article content from the noisy "junk" or unrelated content in web pages has numerous applications. The applications range from information retrieval and indexing to re-purpose and re-formatting the content for better human consumption. A human user is good at figuring out the related parts from unrelated parts. But, a typical automated agent would be implemented to process a web page in entirety with the noisy content mixed with relevant article content. This adversely affects the performance of such automated agents. For example, a naive keyword search can be fooled by unrelated content surrounding articles. Web link analysis approaches can also be influenced by large number of sponsored commercial links or links to other unrelated content. So, separating the main web page content from unrelated content is an essential pre-processing step in improving performance of other web content analysis techniques.

While a human user can intuitively separate out the main content from the "junk" content, often users feel overwhelmed by having to constantly track main content areas and focus away from unrelated "junk". Also, if the users want to print the web page they would prefer to get a clean print out of only the article without printing the unrelated content such as, advertisements or navigation bars. Moreover, for viewing the web content on smaller displays, it may be desirable to re-format the extracted main article content and remove or reduce unrelated content for some applications.

4. ALGORITHMS

4.1 Overview of processing workflow

The processing pipeline is illustrated in Figure 1. The first stage of the system is to extract the article body, which is followed by NLP analysis to find the relevant article images. To summarize, the input to the system is a web HTML page or collection of HTML pages from which the DOM (Document Object Model) Tree is created to extract the various content objects in the page. The DOM objects are analyzed to extract the content blocks that contain the article text body. In the next step, content sub-blocks in the article block are further analyzed to eliminate easy-to-identify unwanted blocks such as lists of links. The remaining content should now be article text body and any image(s) around the article block. Finally, our semantic similarity algorithm based on NLP tries to associate relevant images with main text content based on captions around images. The semantic similarity algorithm also eliminates unrelated images such as advertisements.

Detailed descriptions algorithms are provided below.

4.2 Web Article Text Block Extraction

Every HTML file can be mapped to a DOM (Document Object Model). The DOM is a logical data structure that represents the organization of the various content objects in the HTML file. Because of the embedding nature of HTML, the structure is hierarchical as illustrated in Figure 2. The DOM mapping is necessary for efficient analysis and manipulation of the page content. Each node of the DOM tree is an HTML-Tag element. All text and images that appear on the web page should be a leaf node on the DOM. In our work, we classify each HTML tag element to be either a "block" or "style" element. Each block element, such as the <p>, is rendered into a content block on the web page. It impacts the page layout or the relative positioning of the content. Style elements, in contrast, only affect the visual attributes of the content, such as the font size or color but not the layout. We define block tags to be the following set (all others are considered as style tags): div, p, br, li, ul, ol, td, tr, table, h1-6,hr.

In a typical web article, the text-body is visually separated into paragraphs and contiguously laid out. This visual structure, however, is not always obvious on the DOM. For example, all paragraphs might not be on the same level of a DOM subtree; instead they might be in different subtrees at different level of the DOM. Thus, the challenge of article text-body extraction is to determine how the text blocks (paragraphs) should be clustered together to create the article text-body. There are some recent publications on web article extraction based on DOM analysis[9],[4]. All of these approaches focus on extracting the article text-body, which is simpler than our goal of also extracting the associated image(s). The main challenge here is to differentiate article images from "auxiliary" images, such as ads and navigation menu. One possible approach is to use image analysis technique to extract semantic information from the images directly. This approach, however, is too complicated and the current image analysis technology has not reach the maturity of robust semantic extraction. This work, thus, proposes using the image caption (text around the image) to conduct NLP similarity test between the caption and the text-body. We first describe our method of article text-body extraction.

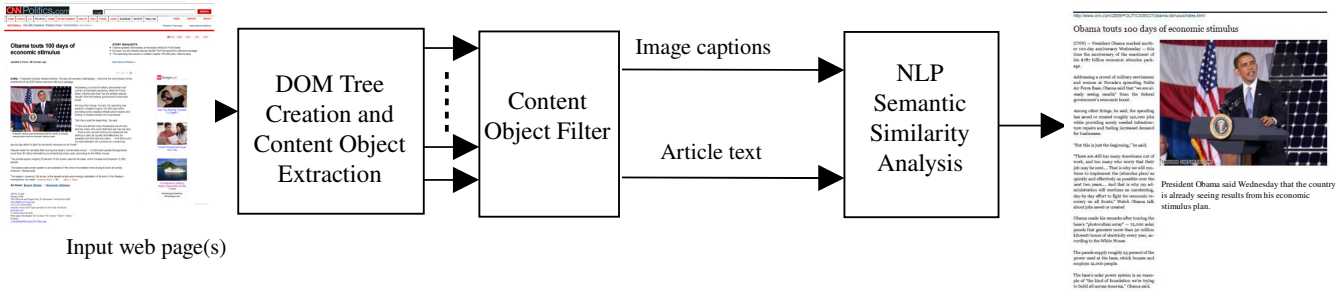

Figure 1: Workflow for web article extraction pipeline

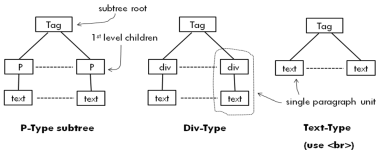

Figure 2: Article sub-tree types

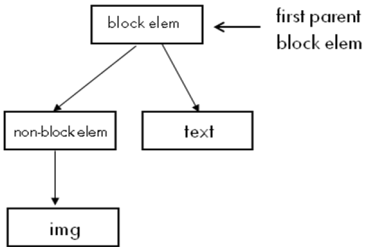

Figure 3: Image and caption sub-tree block

After examining many web article pages, we have found that there are basically three ways in which paragraphs are created in HTML files: (1) text, (2) <p>text, and (3) text
text, as illustrated in Figure 3. Note that both and <p> are block tags where each of these nodes forms a paragraph block on the rendered web page. The third type, text
text, on the contrary uses the
 (line-break) tag to separate the text into paragraph blocks. Figure 3 also shows that the parent node of a set of paragraph nodes at the same level of a subtree is defined as the article subtree. In a typical article, there are usually more than one article subtrees that are candidates as the main article text-body, as illustrated in Figure 2. To determine which article subtree is the main article, we use the following rule-based algorithm. First, find all the article subtrees having large body of text by summing up all the text size (no. of chars) in the paragraphs as defined in Figure 3, then use a threshold to only keep the subtrees with large text-sizes. A good threshold value is 500, which mean the minimum article should contain at least 500 characters. Each article subtree, visually, is a content block on the rendered web page. The article subtree or content block might contain other content type beside paragraphs such as images and hyperlinks to other web sites. If the first step produces multiple article subtrees, then we choose the one that is first occurring on the DOM, which visually should be the uppermost on the web page. Note that this article subtree might not have the largest text-size. The assumption is that the main article text block should always be on top of the page relative to other large text blocks.

4.3 Web Article Image Extraction

Since we are using the image caption for the semantic similarity test, the image and its caption need to be identified and extracted. On the DOM, it is easy to identify images, by simply using the HTML tag semantic, e.g. we look for the <img> tag elements. We assume that if there is an article image, it will be embedded in the article block, equivalent to having the image node in the article subtree. We further assume that if an image has no caption, it is a non-article image. Thus we only consider images with captions in the article subtree as candidates for similarity testing. By examining the DOM of many web pages, we found that the smallest bounding block that contains both the image and its caption is also their first parent block-element, as illustrated in Figure 4. From this observation, the following heuristic rules are used to extract image caption: For each image in the article subtree, find the first parent block-element. All the text content in this parent element is considered to be caption text, which can now be used to conduct the similarity test with the article text-body.

4.4 Web Article Semantic Similarity Evaluation

Text extracted from various text blocks as well as image captions are compared for semantic similarity in this step.

Our content similarity algorithm relies on NLP techniques for information extraction. The algorithm first performs Named Entity Recognition where names of people, places and organizations (commercial, governmental or NGOs) are extracted. To perform Named Entity Recognition we have leveraged GATE[3]. After the Named Entity Recognition is completed, our algorithm compares the extracted patterns of named entities in any two given content blocks to determine content similarity. For smaller content blocks that do not have high frequencies of named entities, it performs match of representative named entities that occur with high frequencies in the bigger content block. For large content blocks our algorithm calculates frequencies of each recognized entity in both content blocks. Then it normalizes frequency distributions and maps normalized frequencies into a multi-dimensional space with number of entities representing the number of dimensions. From that it derives a cosine similarity based on the two normalized frequency distributions from

Figure 4: Example web page with 4 images which are distinguished by semantic similarity

two text blocks. This represents the angle between two frequency distribution vectors. Orthogonal vectors would have cosine similarity measure 0 which represents no similarity. Perfect similarity would be indicated by vectors in the same direction, where cosine similarity measure would be 1.

Cosine Similarity Measure $= \frac{\vec{V_a}.\vec{V_b}}{(|\vec{V_a}|.|\vec{V_b}|)}$ where, $\vec{V_a}$ and $\vec{V_b}$ are vectors representing normalized frequency distributions for the recognized named entities and $|\vec{V_a}|$ and $|\vec{V_b}|$ represent magnitudes of the vectors.

An example web page is shown in Figure 4. Three of the four images are completely unrelated to the main article text. Our algorithm for semantic similarity is able to compare the image captions to the main text content and correctly determines that only the image with President Obama is the relevant image and all other images are unrelated. Final output with all unrelated content removed is shown in Figure 1. It should be noted that since this approach relies on semantic similarity it will work even if any advertising images were placed in middle of the article. Visual layout approaches or pure DOM analysis approaches could be fooled by intelligent positioning of advertisements in the middle of the article as opposed to positioning them in a side bar.

5. CONCLUSIONS

In this paper, we describe a novel approach of using NLP to differentiate relevant images from ad images in web article pages. In general, it is a difficult task to accurately identify informative images simply by analysis of the page layout structure, without any semantic information. We observe that most images in article web pages have textual cues such as caption around the images for which semantic comparison can be performed with the main text body to determine image relevance. The text body and caption extraction schemes are based on DOM analysis, and their accuracy is over 90%. The observed success rate of correctly identifying the informative images is around 83%. The overall processing time is around 5 seconds, including HTML page download in a typical broadband network environment. As for future work, we like to evaluate our system on a large set of web pages. In addition, we like to optimize the content extraction and similarity measure using statistical techniques.

6. REFERENCES

[1] B. Adelberg. Nodose—a tool for semi-automatically extracting structured and semistructured data from text documents. In *SIGMOD '98: Proceedings of the 1998 ACM SIGMOD international conference on Management of data*, pages 283–294, New York, NY, USA, 1998. ACM.

[2] D. Cai, S. Yu, J.-R. Wen, and W.-Y. Ma. Vips: a vision-based page segmentation algorithm. Technical report, Microsoft Research, 2003.

[3] H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan. GATE: A framework and graphical development environment for robust NLP tools and applications. In *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics*, 2002.

[4] S. Gupta, G. Kaiser, D. Neistadt, and P. Grimm. Dom-based content extraction of html documents. In *WWW '03: Proceedings of the 12th international conference on World Wide Web*, pages 207–214, New York, NY, USA, 2003. ACM.

[5] A. H. F. Laender, B. A. Ribeiro-Neto, A. S. da Silva, and J. S. Teixeira. A brief survey of web data extraction tools. *SIGMOD Rec.*, 31(2):84–93, 2002.

[6] S.-H. Lin and J.-M. Ho. Discovering informative content blocks from web documents. In *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 588–593, New York, NY, USA, 2002. ACM.

[7] L. Liu, C. Pu, and W. Han. Xwrap: An xml-enabled wrapper construction system for web information sources. In *ICDE*, pages 611–621, 2000.

[8] L. Ma, N. Goharian, A. Chowdhury, and M. Chung. Extracting unstructured data from template generated web documents. In *CIKM '03: Proceedings of the twelfth international conference on Information and knowledge management*, pages 512–515, New York, NY, USA, 2003. ACM.

[9] J. Pasternack and D. Roth. Extracting article text from the web with maximum subsequence segmentation. In *WWW '09: Proceedings of the 18th international conference on World wide web*, pages 971–980, New York, NY, USA, 2009. ACM.

[10] L. Yi, B. Liu, and X. Li. Eliminating noisy information in web pages for data mining. In *KDD '03: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 296–305, New York, NY, USA, 2003. ACM.