Image extraction from online text streams

A straightforward template independent approach without training

George Adam
Research Academic Computer Technology Institute,
Computer Engineer and Informatics Department, University of Patras
Patras, Greece
adam@cti.gr

Christos Bouras
Research Academic Computer Technology Institute,
Computer Engineer and Informatics Department, University of Patras
Patras, Greece
bouras@cti.gr

Vassilis Poulopoulos
Research Academic Computer Technology Institute,
Computer Engineer and Informatics Department, University of Patras
Patras, Greece
poulop@cti.gr

***Abstract*— In this paper we present an efficient system that processes HTML pages in order to extract the useful images from them. The proposed mechanism is template independent and is focalized on HTML pages that include news articles from major portals and blogs. As useful images we define the pictures that are relevant to the news report. In order to extract the image objects of the article we deconstruct the HTML page to its DOM model and we apply a set of algorithms in order to clean and correct the HTML code, locate and characterize each node of the DOM model and finally keep the nodes that are characterized as useful nodes. The proposed mechanism is applied as a subsystem of peRSSonal, a web tool that is used to obtain news articles from all over the world, process them and present them back to the end users in a personalized manner. The role of the mechanism is to feed peRSSonal's database with digital images for browsing and searching purposes. We present the basic algorithms and experimental results on the efficiency of the proposed implementation.**

Keywords-multimedia extraction; image retrieval; web information extraction, image annotation, web mining

I. INTRODUCTION

The World Wide Web is considered nowadays to be one of the basic sources for information gathering and data searching. More and more people go online in order to fulfill many of their everyday tasks, like news reading from websites. These web sites offer enriched content with multimedia objects, in order to provide better news presentation. Many portals and blogs offer a huge amount of information for reading purposes, but their web pages are often cluttered with distracting features around the body of an article that prevent the user from the actual content they are interested in. These features may include pop-up ads, flashy banner advertisements, unnecessary images, or links scattered around the screen. A solution to the aforementioned problem is a system that is able to omit the information that is of no user interest. The idea is to automatically extract the content that seems to interest the user or delete any content that may distract the internet users. Automatic extraction of useful content from web pages has many applications, ranging from creation of large collections of useful content, to lower bandwidth consumption when applied to proxy servers. The strongest motivation of extracting multimedia objects it is to maintain digital libraries that are able to associate news reports and facts to specific multimedia content. The difficult point in this task is to avoid getting objects that are irrelevant to the parsed web documents, such as advertisements.

There is a large body of related work in content identification and information retrieval that attempts to solve useful content extraction problems using various techniques. S. Gupta, G. Kaiser, D. Neistadt and P. Grimm [5] work with Document Object Model [7] tree and perform content extraction in order to identify and preserve the original data. This method is effective for content extraction but it is not focused on news articles. Thus the output is likely to contain useless content when parsing different pages from web portals.

R. Burget [8] proposes a method of information extraction from HTML documents based on modeling the visual information in the document, which is different from the traditional DOM-based methods. A page segmentation algorithm is used for detecting the document layout and subsequently, the extraction process is based on the analysis of mutual positions of the detected blocks and their visual features. Wang et al. [10] formalize Web news extraction as a machine learning problem and learn a template-independent wrapper using a very small number of labeled news pages from a single site. This implementation is based on the visual features of the web pages and was tested using a training set of 40 pages per website. The results shows that it achieved to extract images and animates with high accuracy, while the news bodies and the extracted news articles remain in the same visual style as in the original pages

Another issue that arises when creating a multimedia extraction system is the tagging of the multimedia objects. Many image annotation systems that are designed to tag images are based on visual processing and operate without any knowledge of the accompanying text. They perform content-based image retrieval, which means that the actual contents of the image are examined.

K. Deschacht and M. Moens [1] are presenting an approach to automatically annotate images using associated text. However, the focus is put only on entities, such as persons and objects which are detected and classified. The results have

shown high precision, tested on Yahoo! [1] news. Another interesting implementation [2], introduces an automatic approach that can effectively extract and annotate semantic knowledge for the web images, utilizing page layout analysis methods and the surrounding context. It was used for JPG image extraction, while the semantic knowledge extraction was separated into people, temporal and geographical information extraction.

Y. Feng and M. Lapata [3] aim to create a database of pictures that are naturally embedded into news articles and propose to use their captions as a proxy for annotation keywords. Experimental results show that an image annotation model can be developed on this dataset alone without the overhead of manual annotation. The examined articles were from BBC[2] news and it was proved that the captions express the picture's content 90% of the time. The images are considered to be usually 203 pixels wide and 152 pixels high and the average caption length estimated to 5.35 tokens.

In this paper we present an extension to the useful content extraction mechanism CUTER [6]. This extension is focused on image extraction from news articles and generally from web pages that are written in article-style (title and body). It manages to extract the images that accompany the article with high precision, based on their position on the webpage, their size and aspect ratio. In addition, the system is able to provide image annotation by correctly identifying the captions that the authors have assigned to them. The image tagging is done performing DOM tree analysis of the webpage. The difference in our implementation is that the image extraction and annotation do not rely on specific web layouts. The most relevant works are based on supervised learning and keep information for every website. In contrast, our approach does not require any training period to operate. Moreover, the mechanism as part of a multi-purpose system is able to utilize "knowledge" that derives from a crawler that is executed before CUTER starts the useful content extraction. Such knowledge is focused mainly on the title of the article (as it is presented in the RSS feed) and can be utilized to efficiently locate the starting point of an article.

The rest of the paper is structured as follows. In the following section we present our motivation and in the third section we present the architecture of CUTER mechanism. In the fourth section we analyze the algorithms utilized within the mechanism. Following we analyze some experimental evaluation that is done in order to present the accuracy of the mechanism and we conclude with remarks and future work on the system.

II. WHY PERSSONAL

The concept behind CUTER as part of peRSSonal [9] is the design of a single web place that offers, in a unified way, personalized and dynamically created views of news deriving from RSS feeds. The internal procedure of the system before an article reaches the end user is: first, all news and articles should be collected in real time from major news portals and blogs. At this stage CUTER is generated as we need to extract the text of the articles from the collected HTML pages. Furthermore, analysis of the extracted text with text pre-processing techniques is applied while categorization and summarization follows. Finally, we present the information to the end user assuring a personalized view, free of useless extra information, fitting in this way the user's demands; fulfilling both their personal profile and their device capabilities. Figure 1 depicts the above procedure. This manuscript focalizes on the module of the system that applies the useful text extraction on the collected HTML pages.

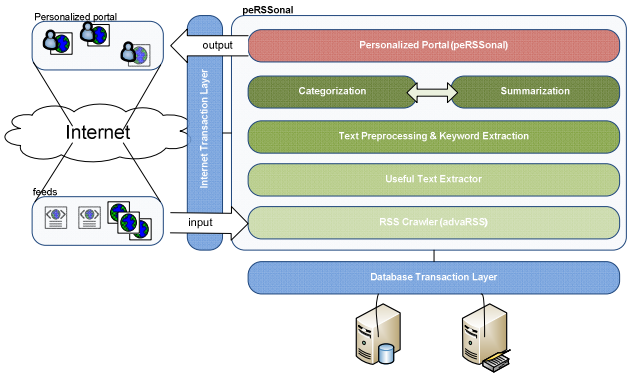

Figure 1. peRSSonal's architecture

III. ARCHITECTURE

CUTER is a standalone system by means of input and output. This makes the mechanism flexible and easily adopted by mechanisms that can feed CUTER with HTML code and can be fed by it with text and multimedia.

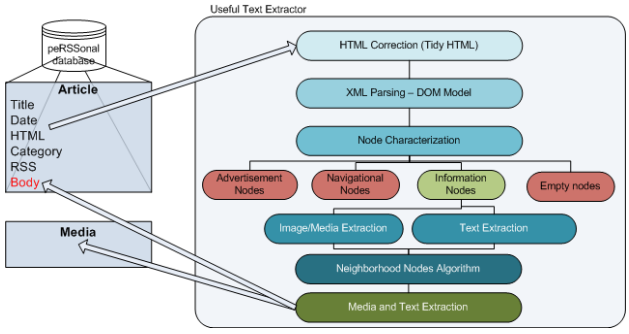

Figure 2. CUTER architecture

Figure 2 presents the architecture of the system. As already mentioned in our introductory part, CUTER is part of a multi-purpose system and thus, some modules of the systems are preceding and some others follow. The system that precedes is an RSS/HTML crawler that is compliant with World Wide Web Consortium[3] standards. It fetches the data of the articles located into RSS feeds and provides the pure HTML code to CUTER.

A first module receives the HTML code from the articles deriving from the RSSs and tries to tidy it. This procedure is essential in order to correct all the HTML tags as all the

[1] http://news.yahoo.com/ - Yahoo! news
[2] http://news.bbc.co.uk/ - BBC news

[3] http://www.w3.org/ - World Wide Web Consortium

browsers allow the HTML programmers to make mistakes without any pay back. The correction will help us create an efficient DOM model of the HTML page. The DOM model is an abstract tree structure of the HTML code with each node of the tree representing the HTML tags. In order to correct the HTML code we utilize HTML Tidy Open Source Project [4].

After the creation of the DOM model, the useful text extraction mechanism starts to process the nodes of the tree in order to characterize the nodes that contain the text of the article. The characterization of the nodes will lead to sets of text nodes which will be processed by the neighborhood nodes algorithm in order to finalize on the exact content that will be extracted by the system. At this stage of the system any nodes that are nested into useful text nodes and include multimedia are characterized as useful multimedia nodes.

IV. ALGORITHM ANALYSIS

Content extraction is a procedure that includes isolation of parts of an HTML page (HTML is referring to the DOM model). In our case the HTML page is a page that includes a single article that may contain images that are important parts of the article. The following figure depicts an article from an online newspaper.

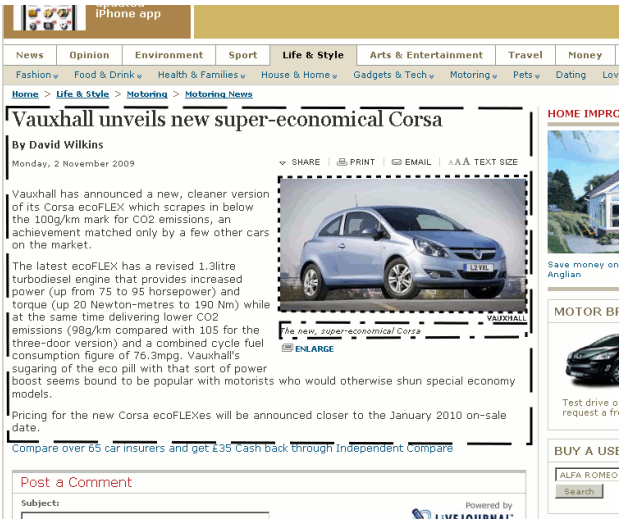

Figure 3. article from The Independent News[4] website

The indicated regions at the Figure 3 are: a) the main article (dashed line), b) relevant picture (dotted line), c) image caption (dash, dot). It is obvious that the useful multimedia objects are located inside the imaginable borders of the article. Images that are outside this area are usually static, as parts of the website's layout, or advertisements.

The first step of the image extractor mechanism is to locate the "borders" of the article, in order to process only images from this region. This task is carried out utilizing the useful text extractor that performs a DOM tree analysis of the HTML code. The output of this procedure includes the nodes that have been characterized as useful text nodes. These nodes correspond to the HTML elements of the original article and are enumerated using a depth-first search. Thus, the borders of the main body can be found by keeping the first and the last node using the aforementioned enumeration.

In many cases, the useful text extractor contains the body of the article, but not its title. This fact can make the system to omit necessary images if they are placed between the title and the starting paragraph, which is very often. A solution is to locate the node that contains the title and to assume that this node is the first node of the article. To overcome the problem that the title is present in more than one HTML elements, the mechanism finds the latest node that contains the title which is placed before the starting paragraph. At the second step, every image located between the first and the last node, is examined based on its dimensions as it is shown in Figure 4.

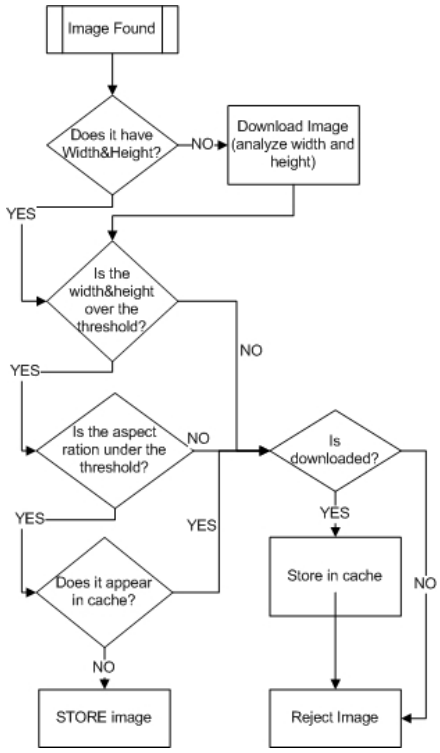

Figure 4. image extraction flowchart

The quickest way to identify the width and the height of the images is to examine the predefined dimensions using the corresponding HTML attributes (width and height). The image is accepted if its width and height are above the appropriate limits and also has an acceptable ratio, which is estimated using the following equation.

$$ratio(w,h) = \frac{\max(w,h)}{\min(w,h)} \quad (1)$$

The average size for images in news portals are about 300px width and 270px height. The system operates with thresholds for the minimum acceptable dimensions that are set to 150px, in order to avoid very small images that are usually placed near the article content for design purposes. The aspect ratio metric is mainly used for advertisement and web banner identification. The banners are considered to be in a high-

[4] http://www.independent.co.uk/ - The Independent newspaper

aspect ratio shape (wide and short, or narrow and tall). It is obvious that lowering these thresholds will result in more accepted images, affecting the precision of the mechanism.

However, the width and height HTML attributes are not always both defined, or are having relative values. In these cases it is unsafe to determine whether the image is useful or not because the information about their dimension is incomplete. Thus, the image has to be downloaded and manipulated by a graphics software library. Downloading multimedia objects is a resource consuming process and the usage of cache appears to be imperative need.

The system exploits a cache that is used for storing the URLs of the previously rejected images utilizing the mathematical concept of a finite set, in order to avoid repeated downloads. It can be implemented by any abstract data structure that can dynamically store certain values, without any particular order, and no repeated values. The URL of an image object is found using a procedure that utilize the "src" HTML attribute and applying the required transformations in relative addressing cases. The cache is initially empty and is extended with a new item, whenever an image has been downloaded and then rejected. These images are usually hosted under a static address in order to be cached by web browsers and web proxy servers. Once the URL is stored in the cache, it can be used in the future by accessing this collection rather than re-fetching the original picture, leading to lower time and bandwidth consumption. The cost of reading the cache is not insignificant, thus the searching operation is not performed in all cases, but only when a picture is going to be downloaded.

The images that cannot be found in cache are downloaded and then examined. Firstly, the mechanism compares the size of the fetched file to a minimum file size threshold, in order to avoid image analysis on very small images that are unlikely to be useful. This threshold is set to 2 kilobytes, taking into account that the average picture for news articles is found to be about 25kb and the minimum about 4kb. A file that is above 2kb is examined using a library for dynamic image manipulation. The system is able to obtain the width and height from the three popular formats for web images: GIF, JPEG and PNG. Having the picture dimensions, the acceptance or the rejection is done using the aforementioned width, height and ratio thresholds, as it can be shown in the Algorithm 1.

Every image that belongs to the article is usually described with a short tag, placed at the bottom of the picture, by the author. The mechanism utilizes the DOM model of the HTML web page, in which each node of the tree representing the HTML tags, in order to locate the image annotation. The extraction of the image caption is based on three observations by examining the news portals from our database. First, the average tag length is expected to be more than 30 and less than 200 characters long. Second, when using depth-first search on the DOM tree, the image node is found to be close and before the tag node. Third, the image and the annotation nodes are not sharing a common ancestor with any other useful text node.

Using the above observations the system starts from the image node and climbs the DOM tree. The searching continues until a close ancestor is found that contains only one node, after the image node, with more than 30 characters and no one with more than 200 characters text. In addition, the system performs a secondary tag extraction procedure, using the "alt" and "title" HTML attributes. This method guarantees that the annotation will be relevant to the picture, but is not always provided or is too short, measured in characters.

ALGORITHM 1. IMAGE EXTRACTION ALGORITHM

```
images[] = Find_Images(startNode, endNode);

Foreach(images[] as image)
   If (Width_Height_Tags_Found(image)) Then
      If(image.width >= WIDTH_THRESHOLD &&
        image.height >= HEIGHT_THRESHOLD &&
        aspectRatio(image)<=RATIO_THRESHOLD)
      Then
         accept(image);
      Else
         reject(image);
      End If
   Else If (isInCache(image))
      Then
         reject(image);
      Else
         imageFile = download(image);
         findRealDimensions(imageFile);

         If (size(imageFile)>=SIZE_THRESHOLD &&
           image.width >= WIDTH_THRESHOLD &&
           image.height >= HEIGHT_THRESHOLD &&
           aspectRatio(image)<=RATIO_THRESHOLD)
         Then
            accept(image);
         Else
            addToCache(image);
            reject(image);
         End If
   End If
End For
```

V. EXPERIMENTAL EVALUATION

In this section we evaluate the performance of the proposed implementation using real news articles from major portals. The evaluation is done by utilizing the precision and recall metrics. The precision is defined as the number of images that are part of the article and are retrieved, divided by the total number of retrieved images.

$$precision = \frac{|\{useful\_images \cap retrieved\_images\}|}{|retrieved\_images|} \quad (2)$$

The recall metric is defined as the number of the articles that are part of the article and are retrieved, divided by the total set of the useful images that are examined.

$$recall = \frac{|\{useful\_images \cap retrieved\_images\}|}{|useful\_images|} \quad (3)$$

The tagging procedure is evaluated by examining if the original correct annotation has been extracted or not. This precision metric is applied only on extracted images that are considered to be useful.

$$tagging\_precision = \begin{cases} 1, retrieved\_tag = original\_tag \\ 0, else \end{cases} \quad (4)$$

The experiment was conducted after collecting and examining 300 random articles from our database in order to estimate the overall precision of the mechanism. In this dataset are included articles that may have more than one or none useful images.

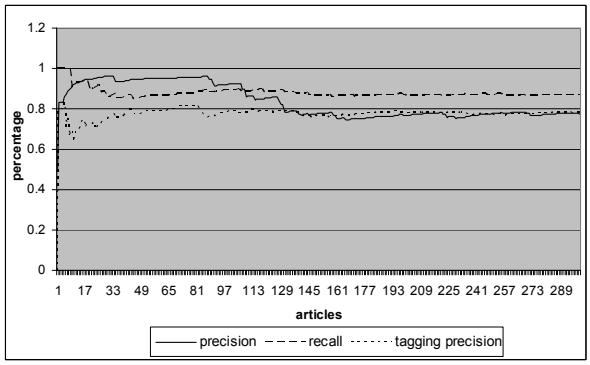

Figure 5. average precision, recall and tagging precision

As it is obvious from Figure 5, the average precision and recall metrics converge after examining more than 170 articles. The final values are about 77% precision and 87% recall (or 81.7% in terms of F1-value) for the image extraction procedure. Precision is being affected by advertisements that are nested into the article's body and are having acceptable aspect ratio and size. They cannot be cached as they are generated dynamic and are having different URL for each article. The recall metric is mainly affected by the usage of CSS [5] and JavaScript features for image rendering. These images are rejected as they are not "near" to the article when performing DOM analysis.

The tagging mechanism achieves to find the correct image description in 78% of the cases. When the original annotation is not found, the mechanism usually extracts a part of the article as image tag. Although this tag is not the correct, it can be used for categorization as it is relevant to the article and maybe relevant to the extracted picture.

The efficiency of the mechanism depends on the layout and structure of the websites that are visited. The precision, recall and tag precision metrics vary when processing different news portals as can be shown in the following figure.

[5] http://www.w3.org/TR/CSS2/ - Cascading Style Sheets

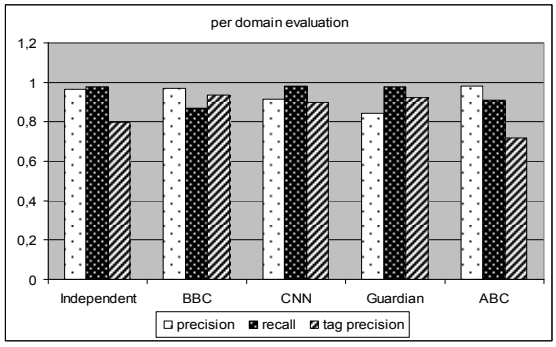

Figure 6. per domain evaluation for 5 portals

Except from the image extraction and annotation procedures, the caching implementation is also evaluated for its efficiency. We use the hit and miss rates when an image is found or not in cache. The following chart shows the results from 10630 examined images when using an initially empty cache.

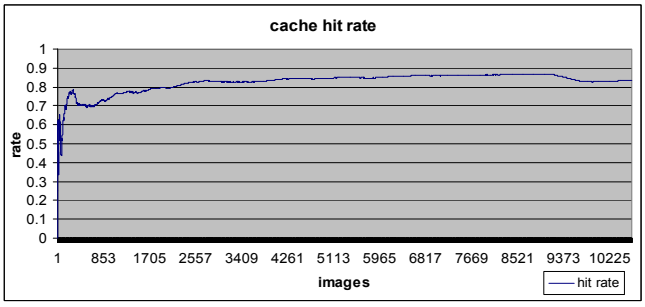

Figure 7. cache hit rate

As it is shown in Figure 7, the rate for images that are found in the cache is about 84%. The caching is done in order to reduce the resource consumption for repeated downloads of the same image. Thus, having 84% less image downloads, will lead to lower bandwidth requirement.

VI. Conclusion and Future Work

We presented a useful image extraction mechanism that is fed with HTML pages including news articles and extracts the related images out of them, based on their position on the webpage, their size and aspect ratio. Moreover, it is able to locate the annotation that was placed by the author with satisfying performance. The main difference from similar information extraction approaches is that the proposed system is template independent and does not require training. The mechanism is an extension of the useful content extractor CUTER, which is part of a larger system, peRSSonal, that implements article's fetching, analysis, text categorization and summarization and finally, provides a collection of multimedia and news, in a personalized way, through a web portal. CUTER is a crucial part of the system as it is the main feeder of text and multimedia to the next procedures and thus its behavior can affect the whole system.

Many improvements can be made to the system at the future in order to increase the efficiency of the image extraction and annotation process. We plan to exploit the visual features

of the retrieved web pages for better handling of sites that use CSS for image positioning. Moreover, a different DOM analysis approach may lead to higher precision and recall. Finally, the caching and retrieving subsystems can be distributed among many peers for load balancing purposes.

REFERENCES

[1] K. Deschacht and M. Moens, "Text analysis for automatic image annotation,". 45th Annual Meeting of the Association of Computational Linguistics, 2007, pp. 1000 – 1007.

[2] H. Zhigang, W. Xiang-Jun, L. Qingshan and L. Hanqing, "Semantic knowledge extraction and annotation for web images," 13th annual ACM international conference on Multimedia, November 06-11, 2005, Hilton, Singapore

[3] Y. Feng and M. Lapata, "Automatic Image Annotation Using Auxiliary Text Information", ACL HLT Columbus, Ohio, USA, 2008.

[4] HTML Tidy Open Source Project, http://tidy.sourceforge.net/

[5] S. Gupta, G. E. Kaiser, D. Neistadt and P. Grimm, "DOM-based content extraction of HTML documents," WWW 2003, 207-214

[6] G Adam, C Bouras and V Poulopoulos, "CUTER: an Efficient Useful Text Extraction Mechanism," The 2009 IEEE International Symposium on Mining and Web (WAM09), Bradford, UK, May 2009, pp. 26 – 29.

[7] DOM, Document Object Model W3C standard, http://www.w3.org/DOM/

[8] R. Burget, "Layout Based Information Extraction from HTML Documents," Ninth International Conference on Document Analysis and Recognition, September 23-26, 2007, p.624-628.

[9] C. Bouras, V. Poulopoulos and V. Tsogkas, "PeRSSonal's core functionality evaluation: Enhancing text labeling through personalized summaries," Data and Knowledge Engineering Journal, Elsevier Science, 2008, Vol. 64, Issue 1, pp. 330 – 345.

[10] J. Wang, X. He, C. Wang, J. Pei, J. Bu, C. Chen, Z. Guan, and W. V. Zhang, "Can we learn a template-independent wrapper for news article extraction from a single training site?," 15th SIGKDD, 2009, pp. 1345-1354.