An automatic wrapper generation process for large scale crawling of news websites

Iraklis Varlamis
Dept. of Informatics and Telematics,
Harokopio University of Athens
Omirou 9, Tavros, Greece
+30210954405
varlamis@hua.gr

Nikos Tsirakis
Palo LTD
Kokkoni Corinthias 20002
Corinthia, Greece
+302103805826
nt@paloservices.com

Vasilis Poulopoulos
Palo LTD
Kokkoni Corinthias 20002
Corinthia, Greece
+302103805826
pv@paloservices.com

Panagiotis Tsantilas
Palo LTD
Kokkoni Corinthias 20002
Corinthia, Greece
+302103805826
pt@paloservices.com

ABSTRACT
The creation and maintenance of a large-scale news content aggregator is a tedious task, which requires more than a simple RSS aggregator. Many news sites appear every day on the Internet, providing new content in different refresh rates, well established news sites restrict access to their content only to subscribers or online readers, without offering RSS feeds, whereas other sites update their CMS or website template and lead crawlers to fetch errors. The main problem that arises from this continuous generation and alteration of pages on the Internet is the automated discovery of the appropriate and useful content and the dynamic rules that crawlers need to apply in order not to become outdated. In this paper we present an innovative mechanism for extracting useful content (title, body and media) from news articles web pages, based on automatic extraction of patterns that form each domain. The system is able to achieve high performance by combining information gathered while discovering the structure of a news site, together with "knowledge" that acquires at each crawling step, in order to improve the quality of the next steps of its own procedure. Additionally, the system can recognize changes in patterns in order to rebuild the domain rules whenever the domain changes structure. This system has been successfully implemented in palo.rs, the first news search engine in Serbia.

Categories and Subject Descriptors
H.3.7 [Digital Libraries]: Collection, System Issues

General Terms
Algorithms, Management, Measurement, Performance, Design, Reliability, Experimentation, Languages.

Keywords
Web crawling, Web Mining.

1. INTRODUCTION
Generation of information and posting on the Internet has become easier than ever. Everyone is able to generate data and post it on the chaotic place that is called World Wide Web. Journalists have exploited this fact and people who would like to create their own place of posting news, articles and opinions. We are putting our focus on this chaotic situation and the need for collecting, analyzing, combining and rearranging all this widespread information. In order to be able to have access to all this kind of information or at least a combination of it, there is strong need for mechanisms that will be able to collect easily the data, which directly refers to crawlers. The problem for crawling of information is researched more and more over the years. We are focusing on data posted as articles, which could mean data from news portals, news agencies, newspapers' websites or even blogs. By defining the kind of data to collect we have the huge advantage of the clear characteristics of the data to be collected. On the other hand crawling articles has a large number of peculiarities.

A major difference between crawling for news and crawling for web content in general, is that in the former case we have to distinguish between content of interest e.g., links to articles, title and body of news articles, images and video that relates to the article, and, noise e.g., advertisements, links to social media sites, user comments. The main property of news sites is that their pages employ a common structure (template), which makes wrapper generation systems popular in this case for extracting news sites content in the absence of RSS feeds. Wrapper generation systems are usually based on visual and structural features of pages. They usually produce delimiter-based expressions, which are then used by crawlers to extract content from pages.

The main problems that arise when wrapper generation systems are examined outside of the crawler scope are that: (a) improperly structured pages may result in the generation of false wrappers, and, (b) the generated wrappers may become outdated when the template of the site changes and the crawler fails to extract content.

In this paper we present a mechanism, which performs large scale crawling of news websites and blogs. The mechanism is based on the automatic generation of a set of wrappers for each site. The proposed mechanism recognizes and characterizes the home page, the category

pages and the article pages being able to distinguish between the peculiarities that follow each of the pages and the continuously repeating patterns of data that usually surround useful content. Links and content are extracted from the news sites in three steps: first, from the homepage; second, from the category pages; third, from the article pages. The wrapper generation process focuses on different structural elements in each case in order to locate components of interest. The main advantages of the proposed wrapper generation process are that it is automatic and self-corrective, and that it allows for creating a self-healing news crawler. Following the machine-learning paradigm, we build upon a set of structural features that help us distinguish the elements of interest in each page and train a classifier in each step.

The main features that allow this are: (a) the wrappers are extracted automatically at page-level and are validated on groups of pages with similar structure (e.g., article pages from the same site), (b) the crawler uses the wrappers to extract content and simple heuristics to verify the success rate of each wrapper. When the wrappers extracted for a site continuously fail, the crawler, in a self-healing attempt, asks for the re-generation of wrappers.

The proposed system can be used for large scale crawling of news sites. It has been already implemented for palo.rs, which is the largest news aggregator in Serbia, indexing thousands of websites and blogs. The overall process is language agnostic and thus can process content in any language. In this paper, we present the results of a smaller scale experiment, in which we employed a set of 95 news sites as input and collected more than 2000 category pages pointing to more than 90000 articles, which have been crawled.

2. RELATED WORK

Manual wrapper extraction approaches require human experts' effort to define patterns on a page and write extraction code for them. Since the maintenance of such wrappers is a tedious and error-prone task, semi-auto extraction approaches were introduced.

Semi-automatic methods induce wrappers from a set of manually labeled pages. The wrappers are sets of rules, which are then used to extract data from pages with similar structure. Many tools induction such as WIEN [6], Stalker [7] or Soft-Mealy [4] use machine learning techniques to induce delimiter based extraction rules from training examples. However, template level wrapper induction has several limitations. First, existing methods cannot correctly process pages that belong to an unseen template. Second, it is costly to maintain up-to-date wrappers for a large amount number of news websites, because any change of a template may invalidate constitute the corresponding

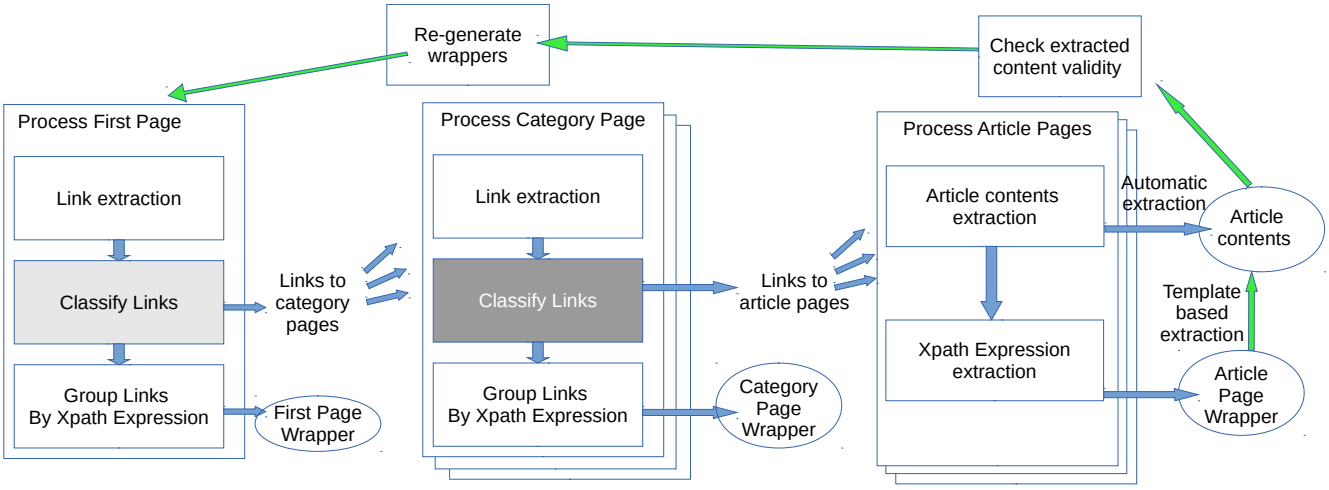

Figure 1. The wrapper generation process

wrapper invalid.

Methods that exploit the visual features of pages [1][13] are very popular for wrapper generation from news sites. They can achieve good accuracy performance since their focus is mainly on the title and body of an article page. However, their computational efficiency is poor because it is time-consuming to render web pages to obtain visual information, and they are not recommended for large scale crawling.

Automatic content extraction methods rely on the extraction of patterns from pages that contain similar data records [12]. They are based on the classification of HTML elements [2][3][5][8], the extraction of a tree that contains the useful pieces of information [10] and tree alignment techniques in order to detect the final template [9][11]. They have good accuracy and do not require visual information or training samples. However, they examine the wrapper generation problem outside the scope of news crawling and cannot be directly applied in a news crawling engine.

The proposed system extends previous works in automatic extraction of news content in the direction of extensively using and testing wrapper generation in a large scale crawling of news sites. It combines the merits of machine learning techniques for tasks that do not require extensive training and of techniques for the automatic extraction of content elements based on structural features. The system provides self-correction for false wrapper generation in a single page, and self-healing capabilities when the crawler finds wrappers that fail to extract content due to template changes.

3. PROPOSED ARCHITECTURE

The proposed crawling mechanism replicates the way a human reader browses a news web site, namely: (a) visits the main page of the news site and follows a category link from the categories bar, (b) browses the respective category page for links to articles of interest and consequently follows these links to open the article page, and, (c) while in an article page, focuses on the article title, the article body and the associated media. The mechanism performs equally well in news sites and blog-structured sites and it is based on a simple hypothesis: The links that lead to article pages differ in structure to the links that lead to category-based or tag-based groupings of blog posts, as well as the fact that crawling should be self-learning and self-healing.

We process pages using a DOM friendly HTML parser (HTMLCleaner - http://htmlcleaner.sourceforge.net/), a set of simple structural features such as the size of text (e.g. in a link or in the article body), the existence of some characteristic extensions (e.g. in links that point to javascript or image files), the position of a link in the DOM tree and a set of node processing modules that take an HTML element as input, check its attributes, its sub or super elements and give a value to the respective structural feature. Table 1 presents some of the structural features that we have been employed.

For simplicity, we assume the three steps browsing process as depicted in Figure 1 and we explain each step separately in the following. However, it is feasible to implement a different crawling strategy, e.g., find all links to articles in any page, starting from the first page, by slightly changing intermediate decisions with regards to which content to keep and which to discard.

3.1 Locate links to category pages

The crawler first processes the site's homepage and searches for links to category pages. The main characteristics of such links are the following: (a) they use a short text in the link (usually a couple of words), (b) they point to pages in the same domain, (c) the relative URLs that they use are also short, and, (d) they are positioned in specific areas of the page (top, bottom, left or right) and (e) they are close to each other, since they are under a common ancestor in the DOM tree and appear as list items in a list or as single contents of consecutive table cells. We formulate all these characteristics as features of a classifier for hyperlinks, using a set of structural features as described before.

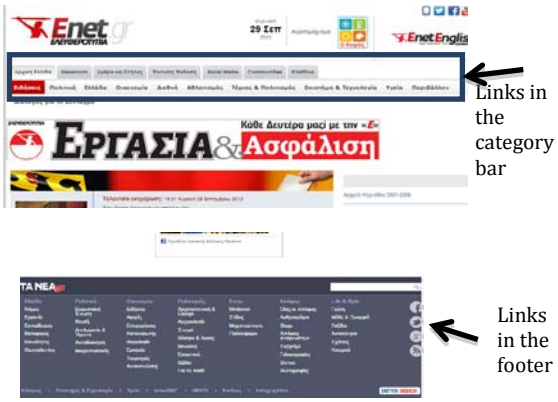

Figure 2. Category links location

The algorithm can easily characterize pages like “about us”, “terms and conditions” or “contact” as category pages, as they appear to have the same characteristics as any category page. In this case by using both page layout and structure rules as well as linguistic rules on the URL of the page we manage to detect most of these not-category pages.

In a first step, we train a classifier to automatically distinguish between links that point to category pages (positive cases) and links that point elsewhere (negative cases). The classifier is trained quickly using a list of news sites’ URLs as input and a human annotator. The training module presents to the annotator a series of randomly selected links, picked from the first pages of aforementioned news sites. The annotator quickly decides whether a link points to category or not usually from the URL only, although, he has the option to check the target page in a browser. During the training process, the classifier’s predictions are validated periodically based on the annotator’s feedback and the annotator is informed on the running accuracy of the classifier. In this way, the annotator can be ready to stop after providing a few hundreds of training samples in less than 10 minutes and reaching an accuracy of more than 90%.

The next step is to use the link classifier in the first page of a web site and locate the area(s) that contain the navigation link. The suggested strategy is to process every single link on the first page, classify it as positive (link that point to a category page) or negative and consequently to group positive links by their location in the page (e.g. see “category bar” and “footer” in Figure 2). For this reason, we use the XPATH expression that locates a link in the DOM tree of the HTML page and group links that share the same XPATH expression. Expressions are build recursively by “climbing up” from the link element to the root of the DOM tree until a unique element (usually or <nav> elements with a unique id attribute is reached). This XPATH expression can be used in future crawler’s visits in order to extract categories from the first page.

3.2 Locating links to article pages

In this task, we follow the same link classification strategy, but with a different set of features that capture the structural characteristics of links to article pages: (a) the text used in the link is longer than in links to categories, (b) the size of the URL is longer too, (c) the links to article pages appear in lists etc. Since a category page may contain many groups of links that point to articles (e.g., Top news, Latest news),

it is advisable for the crawler to use all possible information, which means to extract links using all the suggested XPATH expressions. The crawler also collects the text of the link, which usually corresponds to the article title. As a result, the output of this step comprises the category page wrapper, the article URLs and titles.

Table 1. Selected structural features

Feature	Applies to	Type	Description
EndsWith/ StartsWith/ Contains	Attribute/ Element	Boolean	Checks whether an attribute (e.g. the href of a link) or the text of an element starts/ends/contains a specific string (e.g. .rss, .xml, .js)
ValueLength	Attribute/ Element	Numeric	Computes the length of an attribute value, or an element's text in characters
ValueNumWords	Attribute/ Element	Numeric	Computes the length of an attribute value, or an element's text in words
HasImageChild	Element	Boolean	Checks whether the html element (e.g. an ***<a>*** element) contains an ***<img>*** subelement
HasSiblings	Element	Boolean	Checks whether an element has more siblings (e.g. a link inside a table cell)
IsChildOf	Element	Boolean	Checks whether an element is inside a specific super element
NumChildren	Element	Numeric	Returns the number of direct sub-elements of an element
HostMatch	Attribute	Boolean	Check whether the href of a link points to the same host with the page that contains it
NodesDistanceIn HTMLText	Element	Numeric	Finds the distance of two elements in lines of the HTML text
NodesDistanceIn HTMLDOM	Element	Numeric	Finds the distance of two elements in the DOM tree of the document
NumberOfParagr aphs	Element	Numeric	Returns the number of plain text paragraphs inside an HTML (usually div or nav) element

Figure 3. Article link locations

Every different aspect and view of each page must be evaluated and checked in order to locate links to article pages. The self-learning part of the algorithm prevents multi-collection of the same view of article links across multiple pages. As such, it will not create multiple rules for views that are repeated across pages (e.g. "newsroom") but a single relation of a single page to these specific article links (see Figure 3).

3.3 Wrapper induction for the article page

This step aims in locating the article contents based on structural features, such as the number and average length of sentences (texts/paragraphs) in the article body, the relative position and the proximity between the title and the body, their contextual similarity, the existence of links to media files etc.

The features considered important in this case are summarized to the following: (a) the article body consists of sentences with many words that span a continuous area inside HTML, (b) the title is usually another element which is very similar with the title that we have from crawling the category page, (c) the elements that contain the title and the body are close in the DOM tree, (d) the media files stand either between the title and body or just before their start and end, and the file length is at least a dozen of KBs and finally, (e) we expect common meaning between title and body.

We evaluate these features in order to find the best candidate title element first, then to find the best candidate body with respect to the title, and, finally, to extract the media related to the article (see Figure 4). Text processing techniques are employed in order to remove text

formatting, to link body pieces which are scattered inside the page (e.g., before and after the media) and in order to distinguish between body and the comments.

Figure 4. Article elements

At the end of this process the wrapper generation process results in the candidate XPATH expressions for the body, title and media of each article. The results are aggregated for all the article pages pointed from the same category in order to generalize the findings. The crawler can use the most frequently generated XPATH expressions for future visits in order to avoid the repetitive crawler generation. This can also be of assistance in pages that have a special structure, e.g., pages that contain only a title and a video or image and no content.

As the system holds rules for the structure of the page, which are created automatically, procedures for the system's self-healing and self-correction are easily applied during the process. The self-healing procedure is initiated each time the crawling procedure does not get the expected result based on the existing rules. Additionally, macroscopic evidence is created in order to get information about the performance so as to each micro self-corrections. Each time the average title or body size differentiates from what is recorded throughout the time, the system is ready to apply corrections.

4. EVALUATION

The system is constantly evaluated through the everyday process that is executed in the first news search engine in Serbia, which is done through palo.rs[1]. The accuracy and the quality of results presented in the website are the major proof for the wrapper evaluation. Still, in order to have measurable results we conducted experimental stress tests of the mechanism on 95 different Greek news sites. In this case we obtain much more easily results from the experimental evaluation as the current system for collecting news in palo.gr[2] is based on many years of manual collection of links. The comparison can be direct and measurable.

For our experiments we created a list of 95 news sites in Greece, comprising both blog-structured sites (e.g., sites hosted in wordpress or blogspot) and sites that use their own Content Management System. In the current experimental setup we extract only the area (XPATH expression) that contains the majority of category links. Figure 5 depicts the distribution of sites with respect to the different areas detected by our crawler. As expected, the top and footer bars have been extracted for most of the sites. When a menu bar was missing, category links where extracted by the category tags assigned to the articles.

The whole process returned 2345 category links that point to 2117 distinct category pages. At least one area has been detected in the first page and the average number of areas detected per site was 2.6, which is reasonable since it is common for the first page to contain multiple category bars.

[1] http://www.palo.rs - the first news search engine in Serbia

[2] http://www.palo.gr - the first news search engine in Greece

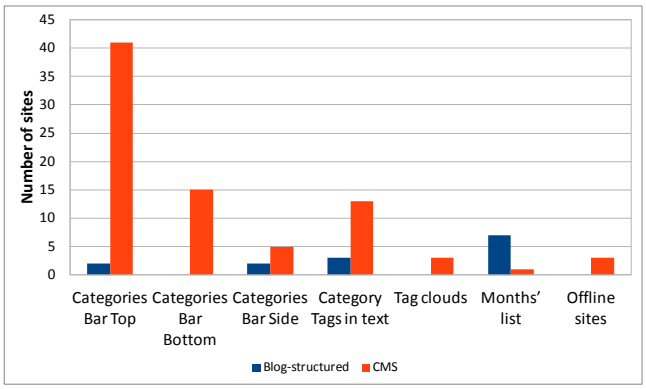

Figure 5. The different areas where category links may appear.

The next step was to scan every category page for links to articles. The distribution of empty category pages checked from each site is depicted in Figure 6. The majority of empty category pages corresponded to empty month entries in 4 blog-structured sites, which means that it was not a wrapper error. Most of the sites (65%) have only one empty category page, which usually corresponds to a "Contact Us" or "About Us" page that contains no articles. This process resulted in 90668 links to articles. In fact the system was able within some minutes to extract rules for a large number of large news portals, blogs and websites and was ready to retrieve articles. Despite, the fact that some methods were used in order to avoid having "fake" category pages we found out that 0.5% of the category links were leading to non-category pages. The only disadvantage is that until some cycles of running for discovering the emptiness in content in these pages and the deletion of them, the crawler will have to parse them.

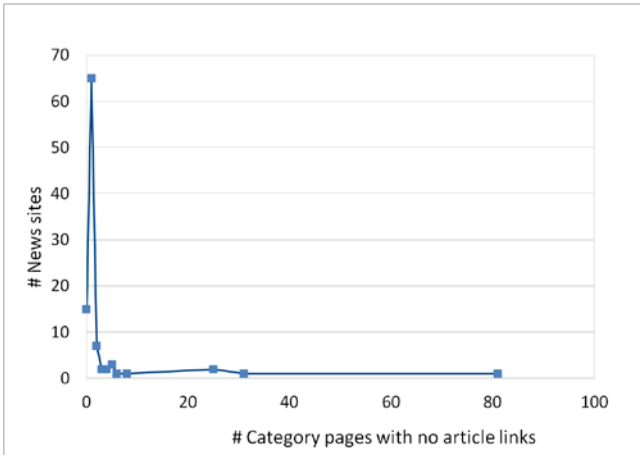

Figure 6. The distribution of empty category pages on the different sites

In the last step, the crawler processed the links to articles grouped by the category pages they have been extracted from. For each group of links, the title, content and media have been extracted along with the respective XPATH expressions for each. The extracted content was automatically checked for validity and the content extraction process was repeated, where necessary, using a different XPATH expression. The result of this process, summarized in Figure 7, was that the crawler failed to retrieve proper content for only a 6% of the articles, which may correspond to articles without textual content (e.g. video or image articles, forms etc). Drilling down in the articles collected by our method we observed a high rate of success as well as a decent percentage of articles that were self-corrected.

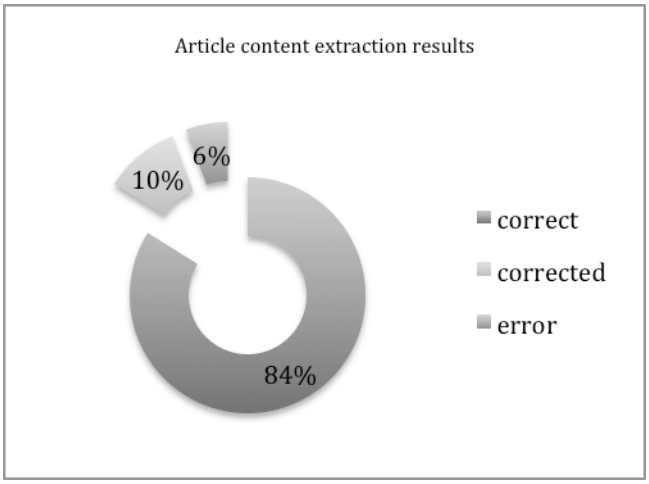

Figure 7. Article content extraction results

For us, it is important to explain the reason for getting errors. In fact, a large percentage of them which reaches almost 40% was related to pages that did not have body at all and was considered as error. The other 3.6%, which was the "real" error, was related to pages with very high complexity of body due to media and ads interference.

Finally, another important factor is the media extraction accuracy of the system, which was also in extremely high levels. The system managed to extract an image in 91% of the cases. 6% of the articles didn't contain media at all, while in 3% of the cases the system was not able to recognize the article media. The reason of failure in most of the cases is related to bad media resolution or aspect ratio, which leads the system to characterize the real images as advertisement images.

5. FUTURE WORK

As the World Wide Web keeps evolving and the content management systems keep updating the need for expanding and making crawling systems more accurate will always be present. In parallel, the research field of data extraction is emerging we keep investigating methods in order to achieve more quality in the procedures of our system. Creating generic rules that could be directly applied to systems that share the same CMS solution is a first approach that could help us reduce the rule discovery procedure time. Additionally, blacklisting or auto-characterization of specific types of pages could help us lessen the retrieval of pages that are not actual category pages.

Nevertheless, the self-healing and self-correction procedures can reduce the errors over time to percentages that are more than adequate for our large-scale crawling system.

6. CONCLUSION

Due to the dynamism of the Web, crawling has become the backbone of any application that provides information retrieval services. In this paper we presented a mechanism for automated extraction of content from news websites and blogs. The mechanism in its final form is fully working on palo.rs website, producing high quality results. This means that the procedures presented and described are fully applied in a large-scale production system with huge success. In parallel we presented the architecture, the procedures and the evaluation of the mechanism.

We evaluated the mechanism and discovered its efficiency when working on large-scale systems that require live data without missing information due to "layout changes". Moreover, our system is self-correcting and self-healing enabling higher accuracy and efficiency.

7. REFERENCES

[1] Cai, D., Yu, S., Wen, J. R., & Ma, W. Y. 2003. *VIPS: a vision based page segmentation algorithm* (p. 28). Microsoft technical report, MSR-TR-2003-79.

[2] Diao, Y., Lu, H., Chen, S., and Tian, Z. 2000. Toward Learning Based Web Query Processing. In *Proceedings of the 26th International Conference on Very Large Databases* (VLDB '00), Amr El Abbadi, Michael L. Brodie, Sharma Chakravarthy, Umeshwar Dayal, Nabil Kamel, Gunter Schlageter, and Kyu-Young Whang (Eds.). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 317-328.

[3] Hossam, I., Darwish, K. and Madany, AR. 2008. Automatic Extraction of Textual Elements from News Web Pages. LREC. 2008.

[4] Hsu, C. N., and Dung, M. T. 1998. *Generating finite-state transducers for semi-structured data extraction from the web.* Information systems, 23(8), 521-538.

[5] Huang, S., Zheng, X., Wang, X., & Chen, D. 2011. News information extraction based on adaptive weighting using unsupervised Bayesian algorithm. In *Web Information Systems and Mining* (pp. 251-258). Springer Berlin Heidelberg.

[6] Kushmerick, N., Doorenbos, R., and Weld, D. 1997. Wrapper Induction for Information Extraction. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 1997

[7] Muslea, I., Minton, S., & Knoblock, C. 1999. A hierarchical approach to wrapper induction. In *Proceedings of the third annual conference on Autonomous Agents* (pp. 190-197). ACM.

[8] Wang JY, Lochovsky FH. 2003. Data extraction and label assignment for Web databases. In *Proceedings of the 12th World Wide Web. Budapest,Hungary*, 2003. PP187-196.

[9] Xia, Y., Yang, Y., Zhang, S., & Yu, H. 2011. *Automatic Wrapper Generation and Maintenance.* In PACLIC (pp. 90-99).

[10] Yan, H., & Yang, J. 2011. A very efficient approach to news title and content extraction on the web. In *Proceedings of the 11th annual international ACM/IEEE joint conference on Digital libraries* (pp. 389-390). ACM.

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

[12] Zaccak, G. 2007. *Wrapster: semi-automatic wrapper generation for semi-structured websites.* Thesis (S.M.)--Massachusetts Institute of Technology.

[13] Zheng, S., Song, R., & Wen, J. R. 2007. *Template-independent news extraction based on visual consistency.* In AAAI (Vol. 7, pp. 1507-1513).