Extracting Information Content from Web Pages Using Block Clustering Method

Nwe Nwe Hlaing, Thi Thi Soe Nyunt
University of Computer Studies, Yangon
nwe2hlaing@gmail.com,ttsoenyunt@gmail.com

Abstract

The World Wide Web is the main "all kind of information" repository and has been so far very successful in disseminating to humans. As web sites are getting more complicated, the construction of web information extraction systems becomes more difficult and time-consuming. Therefore we need to mine the main content of web page in order to extract information from such web pages. In this paper, we study the problem of automatically extracting the web information (unsupervised IE) without any learning examples or other similar human input. Firstly, web pages are segment into several raw chunks. Then remove the noisy blocks based on product features. Data region identification is based on the observation that appearance similarity of the data record in web document. Therefore block clustering method is proposed based on this observation. This approach requires no human intervention and experimental results have shown its accuracy to be promising.

Keywords: Information Extraction (IE), Wrapper, Document Object Model (DOM).

1. Introduction

The explosive growth and popularity of the world-wide web has resulted in a huge number of information sources on the Internet. As web sites are getting more complicated, the construction of web information extraction systems becomes more difficult and time-consuming. Therefore, Web information extraction is an important task for information integration. However, due to the heterogeneity and the lack of structure of Web information sources, access to this huge collection of information has been limited to browsing and searching. Sophisticated Web mining applications, such as comparison shopping robots, require expensive maintenance to deal with different data formats. To automate the translation of input pages into structured data, a lot of efforts have been devoted in the area of information extraction (IE). Unlike information retrieval (IR), which concerns how to identify relevant documents from a document collection, IE produces structured data ready for post-processing, which is crucial to many applications of web mining and searching tools. A typical web page consists of many blocks or areas, e.g., main content areas, navigation areas, advertisements, etc. For a particular application, only part of the information is useful, and the rest are noises. In this approach removing noise based on common web page features. Hence, it is useful to separate these areas automatically.

Web information extraction (WIE) is concerned with the extraction of relevant information from Web pages and transforming it into a form suitable for computerized data-processing applications. Example applications include: price monitoring, market analysis and portal integration. Mining these data records in Web pages is useful because they typically present their host pages' essential information, such as lists of products and services. Many researchers research on extraction of information from web pages in different domains (traveling, products, and business intelligence) but this paper deal with product domain on book store web sites.

It is well known that web pages are used to publish information for humans to

browse, and not designed for computers to extract information automatically. Especially Web pages from shopping site, the underlying structure of current Web pages is more complicated than ever and is far different from their layouts on Web browsers. This makes it more difficult for existing solutions to infer the regularity of the structure of Web pages by only analyzing the tag structures. Meanwhile, to ease human users' consumption of the information extraction, good template designers of Web pages always arrange the data records and the data items with visual regularity to meet the reading habits of human beings. For example, all the data records in Fig. 1 are clearly separated, and the data items of the same semantic in different data records are similar on layout and font.

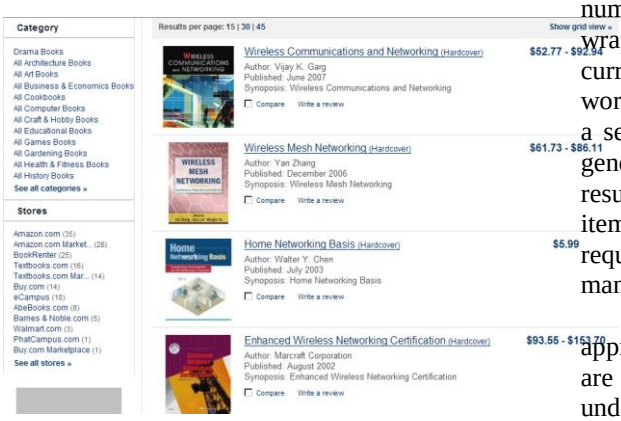

Figure 1. An example Web page from Yahoo book store web site.

Our approach employs a three-step strategy to extract information content on web document. First, given a web page, transform it into DOM tree and parse several initial raw chunks or blocks; second, filtering noisy block based on product features and then clustering remaining blocks which contains all the data records blocks based on their appearance similarity in web page.

This paper is divided into several sections. Section 2 describes the related work on theoretical background and Web information extraction. In Section 3 we discuss our proposed methodology in detail. Section 4 shows the result of our experimental tests while Section 5 concludes this paper.

2. Related Work

Information extraction from web pages is an active research area. The existing works in Web data extraction can be classified according to their automation degree (for a survey, see [5]). There are several approaches [4], [6], [9], [10], [15] for structured data extraction, which is also called wrapper generation. The first approach [9] is to manually write an extraction program for each web site based on observed format patterns of the site. This manual approach is very labor intensive and time consuming. Hence, it does not scale to a large number of sites. The second approach [10] is wrapper induction or wrapper learning, which is currently the main technique. Wrapper learning works as follows: The user first manually labels a set of trained pages. A learning system then generates rules from the training pages. The resulting rules are then applied to extract target items from web pages. These methods either require prior syntactic knowledge or substantial manual efforts.

The third approach [4] is the automatic approach. The structured data objects on a web are normally database records retrieved from underlying web databases and displayed in web pages with some fixed templates. Automatic methods aim to find patterns/grammars from the web pages and then use them to extract data. Examples of automatic systems are IEPAD [4], ROADRUNNER [6], MDR [1], DEPTA [15] and VIPS [3]. Some of these systems make use of the Patricia (PAT) tree for discovering the record boundaries automatically and a pattern-based extraction rule to extract the web data. This method has a poor performance due to the various limitations of the PAT tree. ROADRUNNER [6] extracts a template by analyzing a pair of web pages of the same class at a time. It uses one page to derive an initial template and then tries to match the second page with the template. Deriving of the initial

template has to be again done manually, which is a major limitation of this approach.

Another problem with the existing automatic approaches is their assumption that the relevant information of a data record is contained in a contiguous segment of HTML code, which is not always true. MDR [1] basically exploits the regularities in the HTML tag structure directly. MDR works well only for table and form enwrapped records while our method does not have this limitation. MDR algorithm makes use of the HTML tag tree of the web page to extract data records from the page. However, an incorrect tag tree may be constructed due to the misuse of HTML tags, which in turn makes it impossible to extract data records correctly. DEPTA [14] uses visual information (locations on the screen at which the tags are rendered) to find data records. Rather than analyzing the HTML code, the visual information is utilized to infer the structural relationship among tags and to construct a tag tree. But this method of constructing a tag tree has the limitation that, the tag tree can be built correctly only as long as the browser is able to render the page correctly.

Another similar system is Vints[8]. Vints proposes an algorithm to find SRRs (search result records) from returned pages of the search engines. However, our method focuses on list pages of same presentation template. Although some aspects and pieces of web information extraction may be around in various techniques, the important of this paper focus on the some interesting features of web page and block clustering by appearance similarity.

3. System Overview

This section describes architecture of proposed system. This paper proposes an automatic information extraction from product web pages. The proposed system architecture is shown in figure 2. First of all, input HTML page is changing DOM tree and cleaning useless node as a preprocessing step. Secondly, we tick out several raw chunks as a first round and then filter the noisy block based on noisy features of web pages. Then, thirdly these blocks are clustered by proposed block clustering method. Finally extract information from input web page.

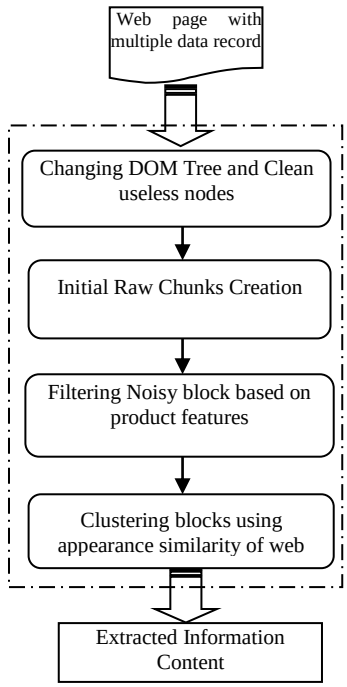

Figure 2. System Architecture

The goal of our system is to offer the extracted data records (shown in Table 1) from web page to the information integration system such as price comparison system and recommendation system.

Table 1. Extracted data record from sample Web page from figure 1.

52.77 92.94	Vijay K. Garg	June 2007	Wireless Communications
61.73 86.11	Yan Zhang	December 2006	Wireless Mesh Networking
5.99	Walter Y. Chen	July 2003	Home Networking Basis
93.55 153.70	Marcraft Corporation	August 2002	Enhanced Wireless Networking Certification

3.1. Features in Web Pages

Web pages are used to publish information to users, similar to other kinds of media, such as newspaper and TV. The designers often associate different types of information with distinct visual characteristics (such as font, position, etc.) to make the information on Web pages easy to understand. As a result, visual features are important for identifying special information on Web pages.

Position features (PFs).These features indicate the location of the data region on a Web page.

PF1: Data regions are always centered horizontally.
PF2: The size of the data region is usually large relative to the area size of the whole page.

Since the data records are the contents in focus on web pages, web page designers always have the region containing the data records centrally and conspicuously placed on pages to capture the user's attention. By investigating a large number of web pages, first, data regions are always located in the centre section horizontally on Web pages. Second, the size of a data region is usually large when there are enough data records in the data region. The actual size of a data region may change greatly because it is not only influenced by the number of data records retrieved, but also by what information is included in each data record.

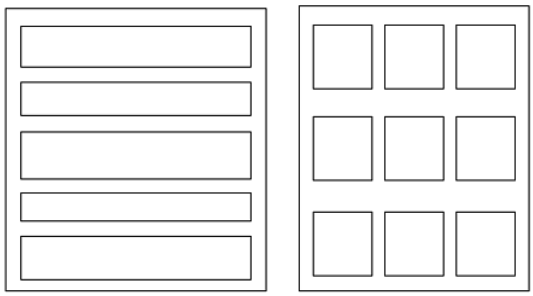

Figure 3. Layout models of data records on web pages.

Layout features (LFs). These features indicate how the data records in the data region are typically arranged.

LF1: The data records are usually aligned flush left in the data region.
LF2: All data records are adjoining.
LF3: Adjoining data records do not overlap, and the space between any two adjoining records is the same.

Data records are usually presented in one of the two layout models shown in Fig. 3. In Model 1, the data records are arranged in a single column evenly, though they may be different in width and height. LF1 implies that the data records have the same distance to the left boundary of the data region. In Model 2, data records are arranged in multiple columns, and the data records in the same column have the same distance to the left boundary of the data region. Because most Web pages follow the first model, we only focus on the first model in this paper. In addition, data records do not overlap, which means that the regions of different data records can be separated.

Appearance features (AFs). These features capture the visual features within data records.

AF1: Data records are very similar in their appearances, and the similarity includes the number of the images they contain and the fonts they use.
AF2: The data items of the same semantic in different data records have similar presentations with respect to position, size (image data item), and font (text data item).
AF3: The neighboring text data items of different semantics often use distinguishable fonts.

AF1 describes the visual similarity at the data record level. Generally, there are three types of data contents in data records, i.e., images, plain texts (the texts without hyper links), and link texts (the texts with hyperlinks). AF2 and AF3 describe the visual similarity at the data item level. The text data items of the same semantic always use the same font, and the image data items of the same semantic are

often similar in size. AF3 indicates that the neighboring text data items of different semantics often use distinguishable fonts.

3.2. DOM Tree Generation and Clean useless node (Preprocessing)

To begin with, a DOM tree should be generated from html tags of the page. At the same time, features/attributes about the node should be included in these tree nodes, respectively; described in next section.

Pre-processing is necessary in order to clean HTML pages, e.g., to remove header details, scripts, styles, comments, hidden tags, space, tag properties, empty tags, etc. In this step, the white relax function of the jsoup parsing tool for removing cleaning HTML tag. First of all, we need to eliminate these nodes to get clean Html page for further processing.

3.3. Initial Raw Chunks Creation

After the generation of DOM tree, following processes are based on it. The root of this tree corresponds to the whole document. The intermediate nodes represent HTML tags (e.g., <table>,<ul>,<tr>,<p>, etc) that determine the layout of the page. At first, those HTML tags, such as <table>, , <tr>, <ul> and <p> naturally form chunks; then we could extract those chunks when encountering them for the first time. That is to say, initially, partition a web page into several raw chunks, just like B-1, B-1-1, B-1-2, B-1-3 and so on illustrated in Figure. 4.

As described in previous section, each node contains some attributes. Here we list these attributes as follows:

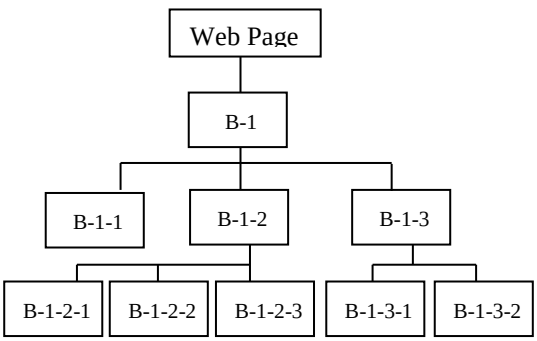

Figure 4. Example of partitioning web page.

{ ImgNum, ImgSize, LinkNum, LinkTextLen, InteractionNum, InteractionSize, FormNum, FormSize, EmailNum}

ImgNum and ImgSize are the number and size of images contained in the block. LinkNum and LinkTextLen are the number of hyperlinks and anchor text length of the block. InnerTextlen is the length of text between the start and ends tags of HTML objects. InteractionNum and InteractionSize are the number and size of elements with the tags of <input> and <select>. FormNum and FormSize are the number and size of element with the tag of <form>. EmailNum is the number of email contained in this block.

Thus, a DOM tree can be parsed into several initial blocks containing relevant features. However, removing some blocks, such as navigation and contact bar are unrelated to the topic at all; therefore for reducing the complexity of information extraction.

3.4. Filtering Noisy Block

Web page designers tend to organize their content in a reasonable way: giving prominence to important things and deemphasizing the unimportant parts with proper features such as position, size, color, word, image, link, etc. All of product page features are related to the importance. For example, an advertisement may contain only images but no texts, a contact information bar may contain email, and a navigation bar may contain quite a few hyperlinks. However, these features have to be normalized by the feature values of the whole page to reflect the image of the whole page. For example, the LinkNum of a block should be normalized by the link numbers of the whole page. Then all these features are formulated with equation (1).

$$fi(a) = \frac{\text{number of attributes in block i}}{\text{number of these attributes in whole page}} \quad (1)$$

Firstly, some conclusions are given on product features. All those conclusions are according to the observation of product list page on web site.

1) If a block contains email elements, then it is entirely possible a contact block.
2) If TextLen/LinkTextLen<threshold, then it is quite possible a hub block [7].
3) If <p> is included in a block, then this block is possible authority block[7].
4) If the normalized LinkNum > threshold, then it is quite possible a hub block.

Accordingly, these rules are calculated into equation (2) and then F indicates the possibility of noisy block.

$$F=\sum \alpha_i.f_i(b)=\alpha_1.f_{email}(b)+\alpha_2.f_{textlen/linktexlen}(b)+\ldots.+\alpha_4.f_{links}(b),\ \sum\alpha_i=1 \quad (2)$$

Where α_i is coefficient, we can set different weights on block importance respectively. Additionally, all these parameters can be adjusted to adapt to different conditions. Also, if F is beyond a predefined threshold, then this chunk has to be ruled out and remove it. Finally regarding product features, an important block is extracted for further processing. Consequently, filtering noisy blocks can decrease the complexity of web information extraction through narrowing down the processing scope.

3.5. Blocks Clustering for Data Region Identification

The blocks in the data region are clustered based on their appearance similarity. Since there are three kinds of information in data records, i.e., images, plain text and link text, the appearance similarity of blocks is computed from the three aspects. For images, we care about the size; for plain text and link text, we care about the shared fonts. Intuitively, if two blocks are more similar on image size, font, they should be more similar in appearance. The appearance similarity formula between two blocks b1 and b2 is given below:

$$sim(b_1,b_2)= W_i .simImg(b_1,b_2) + W_{pt} .simPt(b_1,b_2) + W_{lt} .simLt(b_1,b_2) \quad (3)$$

Where $simImg(b_1,b_2)$, $simPt(b_1,b_2)$, and $simLt(b_1,b_2)$ are the similarity based on image size , plain text , and link text. W_i, W_{pt}, and W_{lt} are the weights of these similarities. Table 2 gives the formulas to compute the component similarities and the weights in different cases.

Table 2. The formulas of block appearance similarity and the weights in different cases

Formulas	Descriptions
$simImg(b1,b2)=\frac{Min\{sa_i(b_1),sa_i(b_2)\}}{Max\{sa_i(b_1),sa_i(b_2)\}}$	$sa_i(b)$is total number of images in block b. $sa_b(b)$ is the total number of block b. $fn_{pt}(b)$ is the total number of fonts of the plain texts in block b. $sa_{pt}(b)$ is the total number of the plain texts in block b. $fn_{lt}(b)$ is the total number of fonts of the link texts in block b. $sa_{lt}(b)$ is the total number of the link text in block b.
$W_i=\frac{sa_i(b_1)+sa_i(b_2)}{sa_b(b_1)+sa_b(b_2)}$	
$simPt(b1,b2)=\frac{Min\{fn_{pt}(b_1),fn_{pt}(b_2)\}}{Max\{fn_{pt}(b_1),fn_{pt}(b_2)\}}$	
$W_{pt}=\frac{sa_{pt}(b_1)+sa_{pt}(b_2)}{sa_b(b_1)+sa_b(b_2)}$	
$simLt(b1,b2)=\frac{Min\{fn_{lt}(b_1),fn_{lt}(b_2)\}}{Max\{fn_{lt}(b_1),fn_{lt}(b_2)\}}$	
$W_{lt}=\frac{sa_{lt}(b_1)+sa_{lt}(b_2)}{sa_b(b_1)+sa_b(b_2)}$	

Our block clustering method consists of two steps: The first one is to build clusters by computing the similarity among blocks. The

similarity $sim(b_1,b_2)$ between two blocks b_i *and* b_j is computed by the equation (3).The second one is to merge the resulting clusters. The threshold is trained from sample page. So the cluster building procedure is simplified as follows:

Procedure BlockClustering

Put all the blocks b_i into the pool;
FOR(every block b_i in pool){
compute the appearance similarity $sim(b_i,b_j)$ bet: two blocks
IF($sim(b_i,b_j)$ >threshold){
group b_i and b_j into a new cluster;
delete b_i and b_j from the pool;
}
ELSE{
create a new cluster for b_i;
delete b_i from the pool;
}
}

The second step is to merge clusters. To determine if two clusters must be merged, we define the cluster similarity $simC_{kl}$ between two clusters C_k and C_l as the maximum value of $sim(b_i,b_j)$, for every two blocks $b_i \in C_k$ and $b_j \in C_l$.

Procedure BlockMerging

FOR(every cluster C_k)
{
compute the $simC_{kl}$ with other clusters;
IF($simC_{kl}$ >threshold){
clusters C_k and C_l are merged;
}
}

4. Experimental Results

Our experiments were testing using commercial book store web sites collected from different web site in Table 3. The system takes as input raw HTML pages containing multiple data records. The measure of our method are based on three factors, the number of actual data records to be extracted, the number of extracted data records from the list page, and the number of correct data records extracted from the list page. Based on these three values, precision and recall are calculated according to the formulas:

Recall=Correct/Actual*100
Precision=Correct/Extracted*100

According to above measurement, we tested web pages from various book store web sites and check each page by manually.

Table 3. Results for selected Web Site

URL	Precision	Recall
gobookshopping.com	100	98
yahoo.com	98	97
allbooks4less.com	100	98
amazon.com	92.4	87.5
barnes&nobels.com	98	90
Average	97.68	94.1

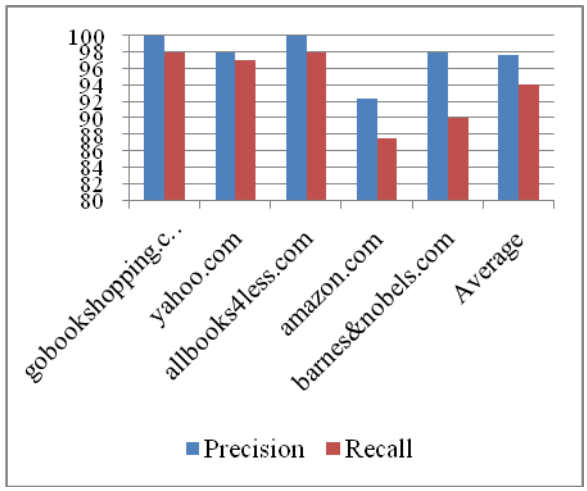

Figure 5. The Precision and Recall Chart for selected web sites

5. Conclusion

In this paper, we have presented extraction of information content from semantic structured HTML documents. It relays on the observation that the appearance similarity of data record in web page. Firstly, we segment a web page into several raw chunks. Second, filter

the noisy block. Then proposed block clustering method groups remaining blocks with their appearance similarity for data region identification. Our method is automatic and it generates a reliable and accurate wrapper for web data integration purpose. In this case, neither prior knowledge of the input HTML page nor any training set is required. We experiment on multiple web sites to evaluate our method and the results prove the approach to be promising.

6. References

[1] B Liu, R. Grossman and Y. Zhai, "Mining Data Records in Web Pages", ACM SIGKDD Conference, 2003.

[2] B Liu and Y. Zhai, "NET – A System for Extracting Web Data from Flat and Nested Data Records", WISE Conference, 2005.

[3] Cai, D., Yu, S., Wen, J.-R. and Ma, W.-Y., VIPS: a vision-based page segmentation algorithm, Microsoft Technical Report.

[4] Chang, C-H., Lui, S-L. "IEPAD: Information Extraction Based on Pattern Discovery", WWW-01, 2001.

[5] Chang, C.-H., Kayed, M., Girgis, M., and Shaalan, K. (2006). "A survey of web information extraction systems", IEEE Transactions on Knowledge and Data Engineering, 18(10):1411–1428.

[6] Crescenzi, V. and Mecca, G. "Automatic information extraction from large websites", *Journal of the ACM*, 2004, 51(5):731–779.

[7] D. Cai, H. Xiaofei, W. Ji-Rong, and M. Wei-Ying, "Block-level Link Analysis", SIGIR'04, July 25-29, 2004.

[8]H. Zhao, W. Meng, Z. Wu, V. Raghavan, C. Yu, "Fully Automatic Wrapper Generation for Search Engines", WWW Conference, 2005.

[9] J. Hammer, H. Garcia Molina, J. Cho, and A. Crespo, "Extracting semi-structured information from the web", In Proceeding of the Workshop on the Management of Semi-structured Data, 1997.

[10] Kushmerick, N, "Wrapper Induction: Efficiency and Expressiveness. Artificial Intelligence", 118:15-68, 2000.

[11] M. Kayed, C.-H. Chang, "FiVaTech: Page-Level Web Data Extraction from Template Pages", IEEE TKDE, vol. 22, no. 2, pp. 249-263, Feb. 2010.

[12] Shian-Hua Lin, Jan-Ming Ho, "Discovering Informative Content Blocks from Web Documents", IEEE Transactions on Knowledge and Data Engineering, page 41-45, Jan, 2004.

[13] Yang, Y. and Zhang, H. "HTML page analysis based on visual cues", In Proceedings of the 6th International Conference on Document Analysis and Recognition, 2001, pages 859–864.

[14] YuJuan Cao, ZhenDong Niu, LiuLing Dai,YuMing Zhao, "Extraction of Informative Blocks from web pages", in the Proceedings of International Conference on Advanced Language Processing and Web Information Technology, 2008.

[15] Y. Zhai, and B. Liu, "Web Data Extraction Based on Partial Tree Alignment", WWW Conference, 2005.