Extraction of Primary Content Blocks from Web Pages

Vinod Ramesh Falmari[1], Ravindra Yallappa Gangundi[2]
[1, 2] Government Polytechnic, Solapur India

Abstract-*A web page can be segmented into different blocks. These blocks define different sections of web page. Each section contains primary or non primary contents. The primary content blocks are considered as required content blocks and the non primary blocks are considered as noisy blocks. The end-users are interested to search for primary content blocks. A tool that searches and processes information from Web pages automatically, must separate the required content blocks from the other content blocks as per user specifications. A user specifies block features to find required blocks. In this paper, we proposed solution methodology that consists of 1) Segmenting Web Pages into Blocks, 2) Identifying Required Blocks based on block features and 3) Clustering of Blocks*

Keywords- Web content mining, Web page blocks, Feature Extraction, Primary content blocks, reverse nearest neighbor

I. INTRODUCTION

Web mining is a rapid growing research area. It consists of Web usage mining, Web structure mining, and Web content mining. Web usage mining refers to the discovery of user access patterns from Web usage logs. Web structure mining tries to discover useful knowledge from the structure of hyperlinks. Web content mining aims to extract/mine useful information or knowledge from web page contents.

Web content mining is related to data mining because many data mining techniques can be applied in Web content mining. It is also quite different from data mining because Web data are mainly semi-structured and/or unstructured, while data mining deals primarily with structured data. The following are research problems in Web content mining:

- Data/information extraction
- Web information integration and schema matching
- Knowledge synthesis
- Opinion extraction from online sources
- Segmenting Web pages and detecting noise

This paper is based on Segmenting Web Pages & Detecting Noise. A typical Web page consists of many blocks or areas, e.g., main content areas, navigation areas, advertisements, etc. It is useful to separate these areas automatically for several practical applications. For example, in Web data mining, e.g., classification and clustering, identifying main content areas or removing noisy blocks (e.g., advertisements, navigation panels, etc) enables one to produce much better results. The research was shown in that the information contained in noisy blocks can seriously harm Web data mining. Another application is Web browsing using a small screen device, such as a PDA. Identifying different content blocks allows one to re-arrange the layout of the page so that the main contents can be seen easily without losing any other information from the page. In recent years, several papers have been published on this topic.

II. RELATED WORK

Deng Cai [8] et al mentioned four types of web page segmentation methods: Fixed-length page segmentation, DOM-based page segmentation, vision-based page segmentation, and a combined method which integrates both vision based semantic and fixed-length properties.

Yi and Liu [1], [2], [6], [7] have proposed an algorithm for identifying noncontent blocks (they refer to it as “noisy” blocks) of Web pages. Their algorithm examines several Web pages from a single Web site. If an element of a Web page has the same style across various Web pages, the element is more likely than not to be marked as a noncontent block. Their algorithm also looks at the entropy of the blocks to determine noncontent blocks. Their technique is intuitively very close to the concept of “information content” of a block. This is one of the very innovative ideas we have studied. In order to identify the presentation styles of elements of Web pages, Yi and Liu’s algorithm constructs a “Style Tree.”

Another work that is closely related is the work by Lin and Ho [9]. The algorithm they proposed also tries to partition a Web page into blocks and identify content blocks. They used the entropy of the keywords used in a block to determine whether the block is redundant.

S.Debnath [1], [2] et al have proposed algorithms for identification of informative sections of Web Pages. Their algorithms look at the Inverse Block Document Frequency and

features of blocks. In order to identify the presentation styles of elements of Web pages algorithm constructs a Document Object Model [3].In the DOM; documents have a logical structure which is very much like a tree. Each document contains zero or one doctype nodes, one root element node, and zero or more comments or processing instructions; the root element serves as the root of the element tree for the document. They have proposed the algorithm FeatureExtractor such that any informative block (corresponding to any feature) can be identified. For example, FeatureExtractor invoked with the features Text, Image, and Links, identifies the text blocks, image blocks, or navigational blocks as the primary content blocks, respectively. The algorithm calculates a value for each feature in each block. After that, put each block in the *winner-basket* if the sum of the feature values of the desired features is greater than the sum of the feature values of the rest of the features. Next from this *winner-basket*, recompute the feature values for this new set of blocks, and chose the one with highest value of desired feature.

For Web pages with multiple important text blocks, a typical reader may be interested in all the sections not just one of them (winner of FeatureExtractor). In k- FeatureExtractor [2], instead of taking just the *winner block* from the *winner-basket*, they applied a k-means clustering algorithm to select the best probability blocks from the basket.

Clustering is an important issue in the analysis and exploration of data. Data clustering is one of the common techniques used in data mining. Clustering consists in discovering natural groups of similar elements in data sets.DJ O'Neil [4] mentioned different methods of nearest neighbor problem. In nearest neighbors problem- Given a set of **n** points and a query point, **q**, the nearest-neighbor problem is concerned with finding the point closest to the query point.

III RECOMMENDED SYSTEM ARCHITECTURE

Figure 1 shows the Block Diagram of Extraction of primary content blocks from Web pages.

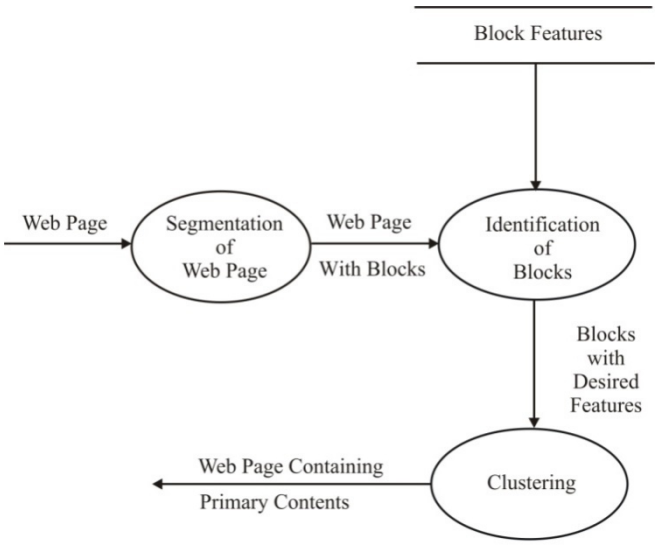

Figure 1. Overall System Architecture

A. Web Page

Web page is simply a HTML page that contains several elements in the form of blocks. The elements are Images, Text, and Applets etc.

B. Desired Block Features

To identify primary content block (corresponding to any feature) FeatureExtractor [1], [2] designed.

For example, FeatureExtractor invoked with the features Text, Image, and Links, identifies the Text blocks, Image blocks, or Navigational blocks as the primary content blocks, respectively.

We can update feaure list if we so desire because of changes in HTML features or because of an application's updated preferences of desirable features easily without fundamentally changing the algorithm.

C. Identification of Webpage Blocks using FeatureExtractor Algorithm

The FeatureExtractor algorithm depends on the feature-set and the chosen feature for output. The set features are HTML features as explained before. For example, let us consider the chosen feature is text (T_1). Now, algorithm calculates a value for each feature in each block.

Say, a block contains 1,000 words and two images and three links and an Applet, and the maximum values of words, Images, links, and Applets contained in blocks in the data set are 2,000, 4, 50, and 3. Then, the values for the

features in the given block are 1000/2000, 2/4, 3/50, and 1/3, respectively. After that, put each block in the *winner-basket* if the sum of the feature values of the desired features is greater than the sum of the feature values of the rest of the features.

The algorithm calculates a value for each feature in each block. After that, put each block in the *winner-basket* if the sum of the feature values of the desired features is greater than the sum of the feature values of the rest of the features.

ALGORITHM 1 : FeatueExtractor

Input: HTML page ***H***, Sorted Tag Set ***T***, Desired Feature F_I
Output: Content Blocks of ***H***
Feature: Feature set $F_S$used for block separation sorted according to importance taken from ***T***
begin
B*←*GetBlockSet (B, T)
{***W*** is the output variable that records potential output block set}
$W \leftarrow \emptyset$
for*each* $b \in B$ **do**
$P_1 \leftarrow Pr(F_1 | F)$
$P_2 \leftarrow Pr((F - F_1) | F)$
if $P_1 > P_2$ **then**
W ← W U b
end if
end for

{Now depending on the condition or choice we will produce output from the set W}
for *each* $b \in W$ **do**
$P_b \leftarrow Pr(F_1 | F, W)$
end for
{Output: Sort ***W*** according to the Probability value P_b}
end begin

In FeatureExtractor, all the blocks of the web page can be available using GetBlockSet. The algorithm compares each block with respect to given features. So the time complexity of FeatureExtractor will be *O (n)* for n blocks.

GetBlockSet

The GetBlockSet Algorithm, shown in Algorithm 4.2, takes an HTML page as input with the ordered tag-set. GetBlockSet takes a tag from the tag-set one by one and calls the GetBlocks routine for each block belonging to the set of blocks, already generated. New sub-blocks created by GetBlocks are added to the block set and the generating main block (which was just partitioned) is removed from the set. The ***First*** function gives the first element (tag) of an ordered set, and the ***Next*** function gives the consecutive elements (tags) of an ordered set.

ALGORITHM 2: GetBlockSet

Input: HTML page ***H***, Sorted tag-set ***T***
Output: Set of Blocks in ***H***

begin
B←H; // set of blocks, initially set to H.
f*←*Next(T)
while*f* ≠ Ø do
b* ← *First(B)
while $b \neq \emptyset$ **do**
if b *contains**f*** then
$B^N \leftarrow GetBlocks(B, f)$ $B \leftarrow (B - b) \cup B^N$
end if
b* ← *Next(B)
end while
f ← *Next (T)*
end while
end begin

GetBlocks

GetBlocks takes a full document or a part of a document, written in HTML, and a tag as its input. It partitions the document into blocks according to the input tag. For example, in case of the <TABLE> tag given as input, it will produce the DOM tree with all the table blocks. It do a breadth-first search of the DOM tree (if any) of the HTML page. If the input tag is <TABLE> and there is no table structure available in the HTML page, it does not partition the page. In that case, the whole input page comes back as a single block. In case of other tags such as <P>, it partitions the page/block into blocks/subblocks separated by those tags. Fig. 3.2 shows the structure of two HTML pages.

D. ***Blocks with Desired Features***

The blocks found by FeatureExtractor algorithm.

E. Clustering of Blocks

1) ***Clustering***

The Clustering problem is concerned with identifying associations between data points. Using the nearest-neighbors algorithm and a maximum distance threshold, the nearest neighbors can be incrementally identified from closest to farthest until the threshold is reached. Clustering is used in

many disciplines, ranging from Marketing to Geography to census data. [4]

2) Nearest Neighbors Problem

Given a set of **n** points and a query point, **q**, the nearest-neighbor problem is concerned with finding the point closest to the query point.

3) reverse-Nearest-Neighbor (rNN):

Given a query point, **q**, rNN finds all points in the dataset such that **q** is their nearest neighbor. The rNN, Algorithm 3, is invoked to form cluster of primary content blocks identified by FeatureExtractor.

ALGORITHM 3: rNN

Input:
***W* is** set of set of probability values P_b from FeatureExtractor
q is query point is the probability value of *winner block* found in FeatureExtractor
N is the number of blocks specified by user
Tn the total number of block found in FeatureExtractor
Output:
C cluster of probability values.

```
begin
        for i ← 1 to N
                d ← MAX ( W [ w_i: Tn ] )
                C ← C U d
        end for
end begin
```

The complexity of the nearest neighbor algorithm actually depends on the number of items. For each loop, each item must be compared to each item already in cluster. Obviously, this is *n* in worst case. Thus the time complexity is $O(n^2)$.

IV.EXPERIMENTAL EVALUATIONS

A. Experimental Setup:

We have implemented FeatureExtractor and rNN algorithms in Java 1.6 on a Pentium-based Windows platform.

B. Results:

We have given various HTML pages and corresponding threshold values of different features. The feature thresholds are such as T number of Text, I number of Images, Tb number of Tables etc. N indicates number of nearest neighbors. The input Data Set and number of blocks found in output are shown in table 1.

C. Comparison with K-FeatureExtractor:

For Web pages with multiple important text blocks, a typical reader may be interested in all the sections, not just one of them (winner of FeatureExtractor). For example, an end-user may be interested in all the reviews available from a page on Amazon.com and each review is in a separate block. General shopping sites, review sites, chat forums, etc., may all contain multiple blocks of important textual information. To overcome this drawback, S Debnath [1] [2] et-al revised the last part of the FeatureExtractor and named the new algorithm as K-FeatureExtractor [2]. Instead of taking just the winner block from the winner-basket, they applied a k-means clustering algorithm to select the best probability blocks from the basket. This helps to get high precision blocks from Web sites.

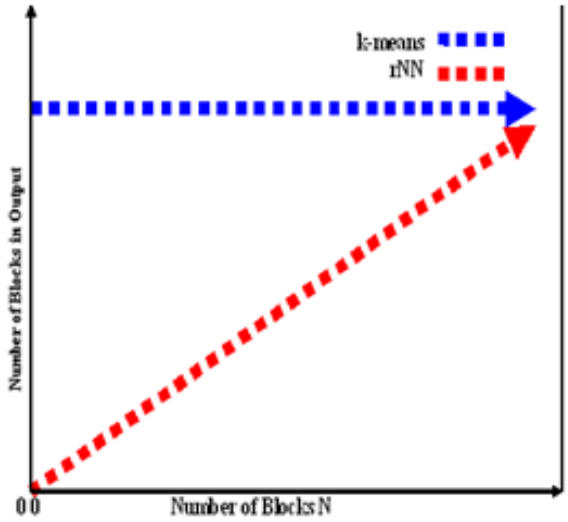

Figure 2. Number of blocks in output after applying k-means and rNN

k-means clustering algorithm takes input the desired number of clusters, *k*, and the set of elements as numeric values. A high degree of similarity among elements in clusters is obtained, while high degree of dissimilarity among elements in different clusters is achieved simultaneously.

In K-FeatureExtractor, the weight (probability value) for each block is found using FeatureExtractor. These weights acts as input set for clustering. The k-means clustering separates all the blocks among k-clusters. The reverse Nearest Neighbor, takes the query point, set of elements and N number of blocks specified by user. We have considered the *winner block* i.e. the output of FeatureExtractor as a query point. The output of rNN contains only N blocks where as k-means

contains all the blocks. The rNN, thereby, reduce the storage requirements, make indices smaller, and it results in faster and more effective search than k-means clustering. The graph in Fig 3 shows the comparison with respect to blocks in output after applying FeatureExtractor then followed by k-means and rNN.

Table1 Sample Input Data Set and Output

Threshold Values								Actual Blocks	Blocks in Output	Memory Saved (%)
T	I	L	F	S	Ls	Tb	N			
22	7	4	2	2	3	2	4	6	4	0.3333
30	2	7	1	3	4	3	2	4	2	0.5000
55	5	11	4	3	0	1	3	1	1	0.0000
40	6	9	2	3	2	4	3	5	3	0.4000
98	5	10	6	2	12	4	7	1	1	0.0000
60	3	4	3	1	7	5	6	9	6	0.3333
76	12	15	3	4	4	5	8	8	8	0.0000

T: Number of Text blocks
I: Number of Images
L: Number of Links
F: Number of Forms
S: Number of Scripts
Ls: Number of Lists
Tb: Number of Tables
N: Number of Nearest Neighbors

V. CONCLUSION

The FeatureExtractor algorithm provided a feature, can identify the primary content block with respect to that feature. The rNN algorithm gives only specified number of primary content blocks as per user specifications. The algorithms, thereby, reduce the storage requirements, makes indices smaller and results in faster and more effective searches make indices smaller, and results in faster and more effective searches. We intend to deploy our algorithms as a part of a system that crawls Web pages, and extracts primary content blocks from it.

REFERENCES

[1] S. Debnath, P. Mitra, N. Pal, and C. Lee Giles, "Automatic Identification of Informative Sections of Web Pages," IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 17, NO. 9, SEPTEMBER 2005

[2] S. Debnath, P. Mitra, and C.L. Giles, "Automatic Extraction of Informative Blocks from Webpages," Proc. Special Track on Web Technologies and Applications in the ACM Symp. Applied Computing, pp. 1722-1726, 2005

[3] World Wide Web Consortium, World Wide Web Consortium Hypertext Markup Language.

[4] DJ O'Neil, Department of Computer Science and Engineering, University of Minnesota "NEAREST NEIGHBORS PROBLEM"

[5] Margaret H. Dunham "Data Mining: Introductory and Advanced Topics"

[6] B. Liu, K. Zhao, and L. Yi, "Eliminating Noisy Information in Web Pages for Data Mining," Proc. Ninth ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining, pp. 296-305, 2003.

[7] L. Yi, B. Liu, and X. Li, "Visualizing Web Site Comparisons," Proc.11th Int'l Conf. World Wide Web, pp. 693-703, 2002.

[8] D. Cai, S. Yu, J.-R. Wen, and W.-Y. Ma, "Block Based Web Search," Proc. 27th Ann. Int'l ACM SIGIR Conf., pp. 456-463, 2004

[9] S.-H. Lin and J.-M. Ho, "Discovering Informative Content Blocks from Web Documents," Proc. Eighth ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining, pp. 588-593, 2002.

[10] Bing Liu, Kevin Chen-Chuan Chang, "Editorial: Special Issue on Web Content Mining"

[11] Hu M and Liu B, Mining and Summarizing Customer Reviews. KDD-04, 2004.

[12] Callan, J. P., Passage-Level Evidence in Document Retrieval, In Proceedings of the Seventeenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Dublin, 1994, pp. 302- 310.