ISOLATING INFORMATIVE BLOCKS FROM LARGE WEB PAGES USING HTML TAG PRIORITY ASSIGNMENT BASED APPROACH

Rasel Kabir, Shaily Kabir, and Shamiul Amin

Department of Computer Science & Engineering, University of Dhaka, Bangladesh

ABSTRACT

Searching useful information from the web, a popular activity, often involves huge irrelevant contents or noises leading to difficulties in extracting useful information. Indeed, search engines, crawlers and information agents may often fail to separate relevant information from noises indicating significance of efficient search results. Earlier, some research works locate noisy data only at the edges of the web page; while others prefer to consider the whole page for noisy data detection. In our paper, we propose a simple priority-assignment based approach with a view to differentiating main contents of the page from the noises. In our proposed technique, we first make partition of the whole page into a number of disjoint blocks using HTML tag based technique. Next, we determine a priority level for each block based on HTML tags priority while considering aggregate priority calculation. This assignment process gives a priority value to each block which helps rank the overall search results in online searching. In our work, the blocks with higher priority are termed as informative blocks and preserved in database for future use, whereas lower priority blocks are considered as noisy blocks and are not used for further data searching operation. Our experimental results show considerable improvement in noisy block elimination and in online page ranking with limited searching time as compared to other known approaches. Moreover, the obtained accuracy from our approach by applying the Naive Bayes text classification method is about 90 percent, quite high as compared to others.

KEYWORDS

Web Page Cleaning, Informative Block, Noise, HTML Tag Priority

1. INTRODUCTION

Data mining has attracted a great deal of attention in recent years, particularly in research, industry, and media. It has gained importance as it required extracting useful information from a wide availability of huge amount of data. In particular, discovery and analysis of useful information or content from the World Wide Web becomes an everyday necessity as raw data is constantly exploding. However, presence of various irrelevant information in the web pages such as banner advertisements, navigation bars, copyright notices may hamper the performance of search engine as well as web mining. Generally, the irrelevant contents in the pages are termed as web noises and are classified as global noise and local noise. Though global noise includes mirror sites, identical pages with different URL, local noise refers to intra-page redundancy (i.e., the irrelevant contents within a page). Noise may mislead the search engine to index redundant contents and retrieve irrelevant results. Moreover, this may hamper the performance of web

DOI : 10.14810/ecij.2015.4305

mining as it extracts patterns based on the whole page content instead of the informative content, which in turn affects negatively in searching results and time. Therefore, removal of such noises is critical for receiving more effective and accurate web mining results.

In this paper, we concentrate on the discovery and removal of local noises from the web pages. We propose a simple algorithm based on HTML tag priority assignment for isolating the main page contents from the noises. In our approach, we first make partition of the whole page into a number of disjoint blocks by using the HTML tag. For each block, we determine a priority level based on the HTML tags priority. Here, we apply the aggregate tag priority calculation. This calculation assign a priority value to each block which helps rank the overall search results in online searching. In our work, the blocks with higher priority are termed as informative blocks and preserved in database for data searching operation, whereas lower priority blocks are considered as noisy blocks and are not used further. Our experimental results show considerable improvement in noisy block elimination and in online page ranking with limited searching time as compared to other known approach *Elimination of Noisy Information from Web Pages* (*ENIW*) introduced in [1].

The rest of this paper is organized as follows. Section 2 reviews previous works on the detection and cleaning of noisy blocks from the web page. This section also pointed out weaknesses in existing literature. In section 3, we introduce our proposed priority assignment based algorithm for noisy block removal. Section 4 presents results from various experiments and section 8 concludes the paper together with future work.

2. LITERATURE REVIEWS

A significant number of researches have been conducted for isolating informative blocks from the noises. Several papers preferred to employ Document Object Module (DOM) tree for block identification. Among them, Cai et al. [3] proposed a vision based page segmentation (VIPS) algorithm that fragments the web page into a number of blocks by using DOM tree with a combination of human visual cues including tag cue, color cue, size cue, and others. Based on the visual perception, it simulated how a user understands the web layout structure. Oza and Mishra [1] proposed a method which extracts the blocks from the page by using Document Object Module (DOM) tree. They considered all the tags of a page as a tree structure. They first removed the tags with no text. Moreover, they discarded the tags unrelated to the page content such as <script>, <style>, <form>, <marquee> and <meta>. Finally, they built the DOM tree structure with the remaining tags. Finally, they identified the location of content from the tree and extracted the content by the html parser. Nevertheless, it is important to mention that the tags <marquee> and <meta> are not contained noisy information for all the time, even if there is no priority value by which they can rank the online searching results. Moreover, they did not consider the ratio comparison between inner and outer hyperlink to detect noisy data. Li and Ezeife [2] proposed a system, WebPageCleaner, for eliminating the noise blocks from the web pages. They applied the VIPS [3] algorithm for the block extraction. By analysing different block features such as the block location, percentage of hyperlinks on the block, and level of similarity of block contents to others, they identified the relevant page blocks. Important blocks were kept to be used for the web content mining using Naive Bayes text classification [7]. However, locating blocks based on the edge of the page is not correct always as at present many article-based websites place the advertisement panel in different template location. Furthermore, Yi and Liu [6] introduced a feature weighting technique to deal with the page noise. They first

constructed a compressed structure tree for getting the common structure and comparable blocks in a set of pages. Next, importance of each node in the compressed tree is estimated by employing an information based measure. Lastly, they assigned a weight to each word feature in the content block based on the tree and its node importance values. The resulting weights are used for noise detection.

Besides, many researchers used the HTML tag information for noise removal from the page. Lin and Ho [4] proposed a method based on the HTML tag <TABLE> to partition a page into a number of content blocks. They calculated the entropy value of each content block by measuring the entropy value of features (terms) present in the block. They dynamically selected the entropy-threshold for partitioning the blocks into informative or redundant. Instead, Bar-Yossef Z. et al. [5] proposed a template detection algorithm based on the number of links in an HTML element. This algorithm partitioned each page into several blocks, called as page-lets, where each page-let is a unit with well-defined topic or functionality. Templates are then detected by identifying duplicate page-lets. However, duplicity of page-lets may mislead the overall work.

3. PROPOSED WORK

Our proposed approach for isolating informative blocks from a large web page is mainly based on the html tag tree. The tag tree is applied to segment the whole page into a number of non-overlapping blocks. Here, a priority value is assigned to each block using different html tags at the block identification time. This priority is later useful for selecting the informative blocks and rejecting the noisy blocks.

3.1. Priority Assignment Based Approached (PAB)

Our proposed priority assignment based (*PAB*) approach takes a number of web pages as input and returns a set of informative blocks for each page. The proposed approach consists of three major steps - Block Extraction, Assignment of Block Priority, and Isolation of Informative Blocks.

3.1.1 Block Extraction

In this step, a web page is divided into multiple blocks by using the html tag tree [8]. To identify blocks, some important html tags are applied. The important tags are - Title (<title>), Head(<H1, H2, H3, H4, H5, H6>), Paragraph (<p>), Bold (<b>), Strong (), Hyperlink (<a>), Italic (<i>), and etc. It is noted that these tags are referred by the Search Engine Optimizers (SEO) [9]. Besides them, some medium level html tags are also utilized for detecting blocks. In our work, Table (<table>) and List (<li>) are considered as the medium level tags. Moreover, some html tags such as <script>, <style>, <iframe>, <object>, and <form> are considered as irrelevant tags. All contents between these irrelevant tags are removed. This operation eliminates a large amount of unnecessary data, which we term as noisy data. Figure 3.1 presents the segmentation of the page into a number of blocks using the html tags in our approach. In block identification, our approach first selects the block containing only the Title (<title>) tag for the page. Subsequently, a sequential searching is performed from the Body (<body>) tag to choose the next available tag <T>. Now, consider all the contents as a single block before the closing of <T>, i.e. </T>. All other tags encountered before </T> are considered as inner tags. By following this technique all the blocks of the page are identified. It is important to mention that if no tag is found between <body> and </body>, then one single block is considered for the whole page.

3.1.2 Assignment of Block Priority

The extracted blocks are given a priority value in this step. For priority calculation, our approach generates a html tags priority table (PT) for the blocks which is based on SEO [10]. In PT, a category label and a priority value are heuristically assigned to the important tags. The Table 3.1 is useful for measuring the priority values for the blocks. Generally, title matching with the user search query implies that the page contents meet the user requirement. Therefore, our approach automatically gives a priority of 1.0 for the title block. In addition, for other blocks a priority value is assigned to each block by accumulating the priorities of all inner and outer tags. Eq.(1) is used for the block priority aggregation. For any block $bi = \{t_1, t_2, ..., t_n\}$.

$$priority\ (bi) = \sum_{j=1}^{n} priority\ (ti) \qquad \text{Eq.(1)}$$

Where t_j represent the tag contained in b_i. Moreover, the proposed method gives a priority of 0.0 to all blocks having more than 50% of outer hyperlinks.

3.1.3 Isolation of Informative Blocks

Our *PAB* approach removes all the blocks having the priority value 0.0 from the block list of each page. These blocks are termed as the noisy blocks. The remaining blocks are kept as database records for using in online data searching and data mining operations. These blocks are termed as the informative blocks.

Figure 3.1 depicts a number of page blocks marked by using the html tag tree. The priority of each block can be measured by applying the tag priority mentioned in Table 3.1. For Block-1 containing Header (<h1>) tag, a priority value of 1.0 is assigned. For Block-2, we can see that this block contains 10 hyperlinks. Though this block starts with the Bold (<b>) tag, but all hyperlinks are belong to other websites. In other words, if the total number of hyperlinks is 10 and the total outer links is 10, then the block contains 100% outer links. Therefore, Block-2 is directly set to a priority of 0.0. Again, Block-3 contains inner tags. This Block is starting with <p>, followed by two inner tags <b>. Thus, the priority value of Block-3 is 0.1 + 0.4 + 0.4 = 0.9, where 0.1 for <p> and 0.4 is for <b>. In a similar manner, the priority value of Block-4 is set to 0.7. Table 3.2 presents the priority of all the blocks identified in Figure 3.1.

Table 3.1. Html Tags Priority Table (PT)

SL	HTML Tags	Category level	Priority value
1	<TITLE>, <H1>	Cat-1.0	1.0
2	<H2>	Cat-0.9	0.9
3	<H3>	Cat-0.8	0.8
4	<H4>	Cat-0.7	0.7
5	<H5>	Cat-0.6	0.6
6	<H6>	Cat-0.5	0.5
7	Bold (<b>), Strong ()	Cat-0.4	0.4
8	Image with alternative text <img>	Cat-0.3	0.3
9	Hyperlink (<a>), italic	Cat-0.2	0.2
10	Paragraph (<p>)	Cat-0.1	0.1
11	Preserve for Noisy Block	Cat-0.0	0.0

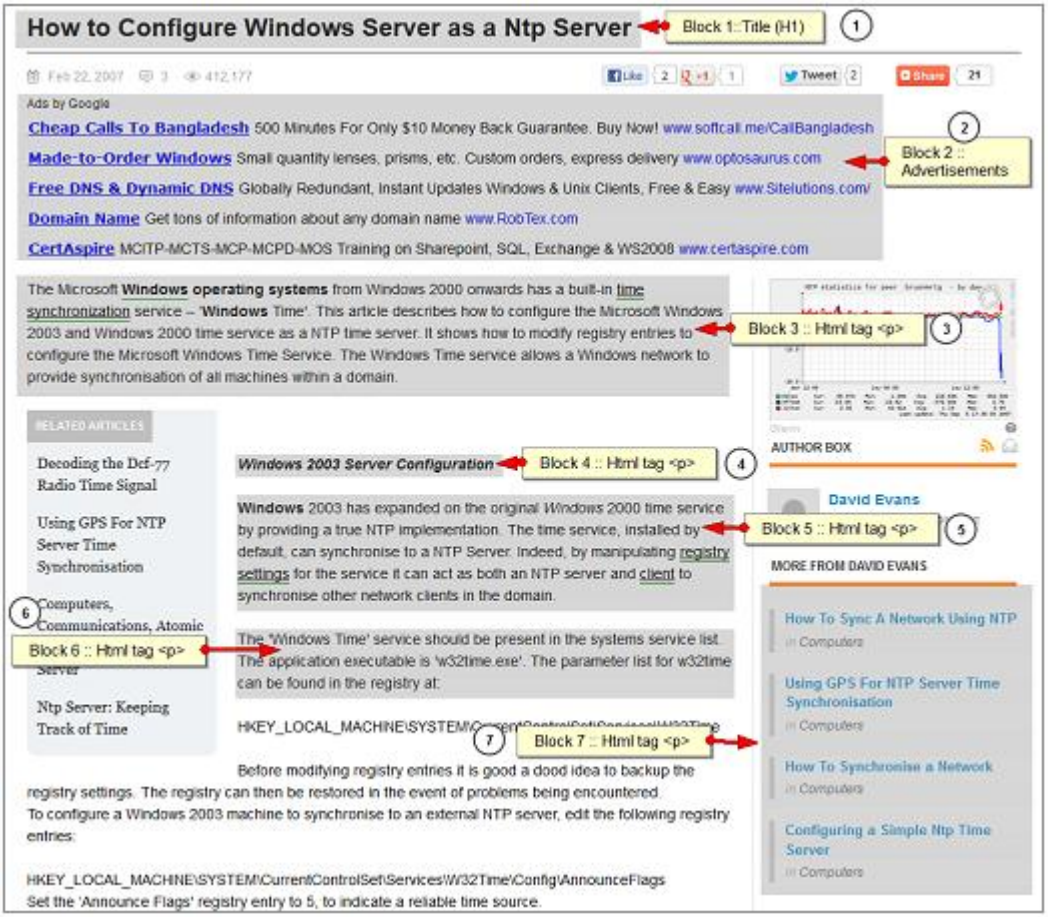

Figure 3.1. A Sample Large Web Page with Multiple Blocks and Advertisement

Table 3.2. Priority Tag (PT) Table

SL	Block number	Tags between block	Priority values
1	Block-1	<H1>	1.0
2	Block-2	<b>,<a> but 100% outer links	0.0
3	Block-3	<p>, <b>, <b>	0.1+0.4+0.4 = 0.9
4	Block-4	<p>, <i>, <b>	0.1+0.2+0.4 = 0.7
5	Block-5	<p>, <b>	0.1+0.4 = 0.5
6	Block-6	<p>	0.1 = 0.1
7	Block-7	<p>, <b>,<a>, 0% Outer Links	0.1+0.2+ 0.2*(4) = 1.1

3.2. Proposed PAB Algorithm

Our proposed *PAB* algorithm has two phases. The first phase *SetPriorityLevel( )* is taking a set of web pages as input. By using the html tag tree, it divides the page into a number of blocks. It also performs the first phase noisy data elimination by removing all unnecessary contents between the irrelevant tags. The second phase *DetectSecondNoisyBlock( )* identifies the second phase noisy blocks based on the priority value. For priority measure, it aggregates all inner and outer html tag priority. Note that the complexity of PAB algorithm is $O(p*b*t)$, where p is the number of pages, b is the number of total blocks per page, and t is the total html tags per block.

Algorithm 3.2.1: *SetPriorityLevel( )*

Input: A set of web pages

Output: An informative block database for the web page

Begin

1. Apply HTML tag tree to the pages and extract block features.
2. For each web page, do steps a, b and c
   a. Remove all contents between the tags <script>, <style>, <iframe>, <object> and <form>.
   b. Call *DetectSecondNoisyBlock( )* to detect and remove the second phase noisy blocks.
   c. Store all informative blocks with higher priority in a block database.
3. Return an informative block database for the pages.

End

Algorithm 3.2.2: *DetectSecondNoisyBlock( )*

Input: A set of blocks for a page and a priority table (PT).

Output: A set of informative blocks

Begin

1. Check the title block and assign a priority value of 1.0 by using PT.
2. Find all inner and outer tags of each block and calculate the priority value for the block by aggregating tags' priority.
3. For all hyperlinks in a block, calculate the ratio of inner and outer links (in %). If the outer links is more than 50% of the total links, then the block is consider as a noisy block and its priority value is set to 0.0.
4. All blocks with priority value greater than 0.0 are considered as informative blocks.
5. Merge all duplicate informative blocks.

End

4. EXPERIMENTAL RESULTS

All experiment were performed on an AMD FX(tm)- 4100 quad-core processor each capable of processing up to 3.60 GHz. The system also had a 6 GB RAM with Windows 7 64-bit operating system. NetBeans IDE 7.3.1 is used as a tool for implementing the proposed *PAB* and the existing *ENIW* algorithm. The two algorithms are implemented in PHP language.

For the experiments, we used 500 distinct web pages downloaded from the commercial article web portal http://www.ezinearticles.com. The collected pages were categorized into 3 major groups. They were *Arts and Entertainment*, *Computers and Technology*, and *Food and Drink*. Details of distribution of the pages are shown in Table 4.1.

Table 4.1. Collection of web pages used in the experiment

Website	Arts and Entertainment	Computers and Technology	Food and Drink	Total Web Pages (Experimented)
www.ezinearticles.com	200	200	100	500

We evaluated the performance of *PAB* with respect to the following parameters: (1) No. of extracted informative blocks, (2) Total size of informative block-database (3) Accuracy measure based on Naive Bayes Text Classification method, (4) Standard Error (SE), (5) Online search result based on a user search query and search time in second, (6) Average searching time in second.

4.1. Number of Extracted Informative Blocks

After applying our *PAB* approach for informative block extraction, we got a total of 120,237 blocks of which 111,918 blocks are informative and the remaining 8,319 blocks are noisy. In case of *ENIW* approach, we got 113,624 informative and 6,613 noisy. The comparison between *PAB* and *ENIW* based on the total number of blocks, extracted informative blocks, noisy blocks and their total time (sec) for informative block extraction is given in Table 4.1.1.

Table 4.1.1. Number of blocks comparison between *PAB* and *ENIW*

Total Page	Total Blocks	Approach	Total Extracted Informative Blocks	Total Eliminated Noisy Blocks	Total Time for block extraction (in sec)
500	120,237	PAB	111,918	8,319	89.56
		ENIW	113,624	6,613	95.58

4.2. Total Size of Informative Block Database

Database size is one of the most important factors for the online search engine. A large database not only involves a huge storage space, but also needs a longer searching time. As the user wants to see the searching results as quickly as possible based on their search query, large sized database directly affects the performance of the search engine in a negative way. For experimenting purpose, we have used 500 articles from http://www.ezinearticles.com. After applying our *PAB* approach, we got 111,918 informative blocks which contains 19.4 MB of the database size, whereas for the *ENIW* we got 113,624 informative blocks involving 22.6 MB of database size. Figure 4.2.1 shows the comparison of database size between *PAB* and *ENIW*.

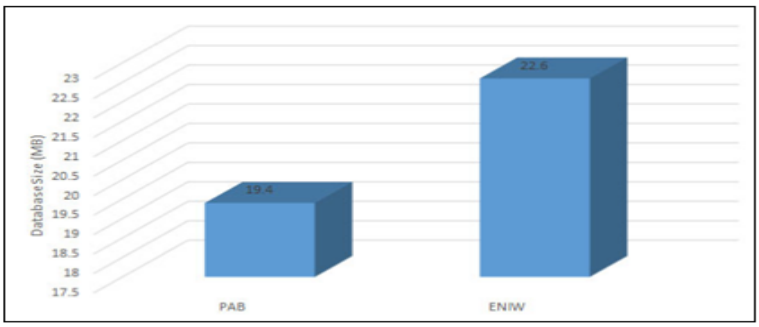

Figure 4.2.1. Database size comparison between PAB and ENIW

4.3. Accuracy Measure based on Naive Bayes Text Classification Method

To compare the accuracy of *PAB* and *ENIW*, we applied the *Naive Bayes Text Classification* method [7]. This text classification, a probabilistic learning method is able to detect the category of hidden test set depending on different training set in various category. Here, the accuracy is calculated by using the following equation:

$$\text{Accuracy} = \frac{total\ number\ of\ correct\ classifications}{total\ number\ of\ classification\ s} * 100\% \qquad \text{Eq.(2)}$$

From a total of 500 pages, we have randomly generated 7 different combinations of the training and the test datasets in order to ensure that the accuracy of category identification is not sensitive to a particular partition of datasets. The random combinations are given in Table 4.3.1. Besides, the percentage of accuracy using *Naive Bayes text classification* method for *PAB* and *ENIW* approaches is shown in Table 4.3.2.

Table 4.3.1. Random combinations of the training and test datasets

Case	Training pages	Total pages
1	490	10
2	480	20
3	470	30
4	460	40
5	450	50
6	440	60
7	430	70

Table 4.3.2. Percentage of accuracy based on *Naive Bayes text classification* method

Case	Test pages	ENIW	PAB
1	10	80%	80%
2	20	85%	80%
3	30	83%	83%
4	40	85%	86%
5	50	86%	88%
6	60	83.3%	88.3%
7	70	80%	90%

From Table 4.3.2, we can observe that for Case-1, *Naive Bayes text classification* method detects 8 corrected web pages from randomly selected 10 hidden test pages for both *ENIW* and *PAB*. For Case-2, it correctly categorises 15 web pages for *ENIW* and 16 web pages for *PAB* in case of 20 randomly selected hidden test pages. For Case-3, it correctly categorises 25 web pages for both *ENIW* and *PAB* in case of 30 randomly selected hidden test pages. For Case-4, it correctly classifies 34 web pages for *ENIW* and 35 web pages for *PAB* in case of 40 randomly selected hidden test pages. For Case-5, it correctly catalogues 43 web pages for *ENIW* and 44 web pages for *PAB* in case of 50 randomly selected hidden test pages. For Case-6, it correctly categorises 50 pages for *ENIW* and 53 pages for *PAB* in case of 60 randomly selected hidden test pages. For Case-7, it correctly categorises 56 web pages for *ENIW* and 63 web pages for *PAB* in case of 70 randomly selected hidden test pages. Figure 4.3.1 presents the percentage of accuracy of *PAB* and *ENIW* based on the seven distinct test sets.

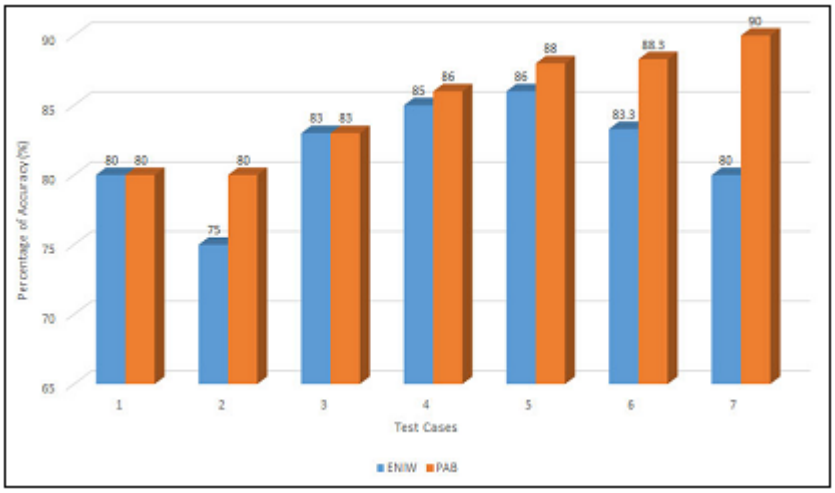

Figure 4.3.1. Average accuracy based on Naive Bayes Text Classification method

4.4. Standard Error

Low standard error (*SE*) presents high accuracy. If the variance represented by sigma square (σ^2) then the SE is represent:

$$SE = \frac{\sqrt{\sigma 1^2 + \sigma 2^2 + \cdots + \sigma n^2}}{n} \qquad \text{Eq.(3)}$$

By using Eq.(3), the comparison of *SE* between *PAB* and *ENIW* is shown in Table 4.4.1.

Table 4.4.1. Comparison of standard error between *PAB* and *ENIW*

Approach	Standard Error
PAB	0.1496
ENIW	0.20396

4.5. Online Search Result based on User Search Query

We have compared *PAB* with *ENIW* and also with real web page data which we called raw data with respect to the online search results. In *PAB*, we discarded more noisy data and as a result, we received a lower number of informative blocks as compared to ENIW. However, the raw data is a mixture of the noisy and informative blocks. For comparing the three approaches based on the user search query, we applied a search keyword *Computer*.

Table 4.5.1. Comparison of *Online Search Results between PAB*, *ENIW*, and *Raw data for the search query Compute*.

Keyword	Approach	Searching time (in second)	Total Page Link Found in Search Result
Computer	*Raw data*	0.278	140
	ENIW	0.235	137
	PAB	0.012	136

Table 4.5.1 presents the searching time and search result for *Raw data*, *ENIW*, *and PAB*. Here, the keyword density is used to sort the searching results in case of *Raw data* and *ENIW*. However, in case of *PAB* we applied the block priority as well as the density of the keyword to calculate sort the searching. Eq.(4) shows the blocks priority calculation of *PAB*.

$$Bp = Pb * Kd \qquad \text{Eq.(4)}$$

where, B_p is the block priority, P_b is the total tag priority value of the block and K_d is the keyword density of that block. Though from Table 4.5.1, we can observe the similar number of page link in the search results, but their searching time are varied. Moreover, the searching time of our *PAB* is very small as compared to both the existing *ENIW* technique and *Raw data*. So for online user, it is very efficient to get accurate result within very short time in a large database using our proposed *PAB* algorithm.

4.6. Average Searching Time in Second

We performed searching based on the 50 randomly selected keywords available in our searching database. We found that the average searching time of *PAB* is significantly small as compared to both the *ENIW* and the *Raw data*. Figure 4.6.1 shows the average searching time comparison among *PAB, ENIW* and R*aw data*.

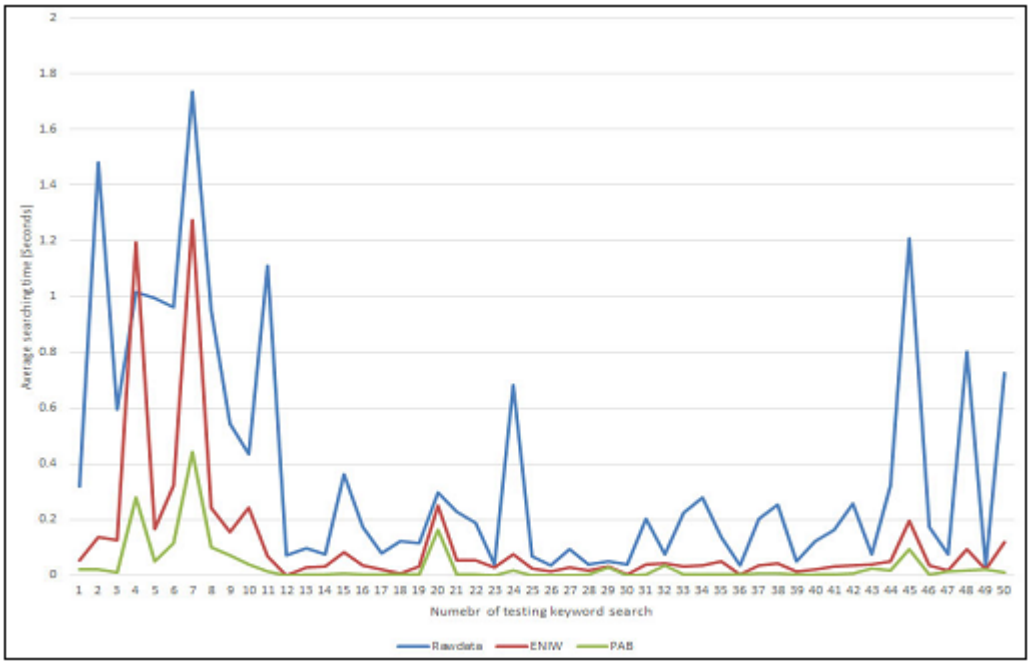

Figure 4.6.1 Average searching time (in second) comparison among *PAB, ENIW* and *Raw data*

5. CONCLUSION AND FUTURE WORKS

Extraction of useful information and removal of noisy data is well-known in Web Content mining. Despite many researches, weaknesses remain in existing algorithms to get rid of noisy data. This paper intended to provide an improved solution to this end. The proposed *PAB* method is developed to remove noisy blocks with a high accuracy when searching and sort them according to their URL ranking. Our *PAB* algorithm scans the whole page through two passes. In first pass, it partitions full page into multiple blocks. Later, it removes some blocks that are under the pre-defined html tags as noisy block. Experimental results show considerable improvement in noisy block elimination and in online page ranking with limited searching time from our *PAB* algorithm as compared to other known approaches. In addition, the accuracy obtained from our *PAB* by applying the *Naive Bayes text classification* method is about 90 percent, quite high as compared to others. Our future research work aims to go further. We intend to group similar informative blocks in a large web page by using clustering method which is likely to improve experimental results in online searching.

REFERENCES

[1] Shailendra Mishra Alpa K. Oza. Elimination of noisy information from web pages. In International Journal of Recent Technology and Engineering, volume 2, pages 574{581, Piplani-BHEL, Bhopal India, 2013. IJRTE.

[2] Jing Li and C. I. Ezeife. Cleaning web pages for effective web content mining. In Proceedings of the 17th International Conference on Database and Expert Systems Applications, DEXA'06, pages 560{571, Berlin, Heidelberg, 2006. Springer-Verlag.

[3] Deng Cai, Shipeng Yu, Ji rong Wen, Wei ying Ma, Deng Cai, Shipeng Yu, Ji rong Wen, and Wei ying Ma. 1 vips: a vision-based page segmentation algorithm, 2003.

[4] Shian-Hua Lin and Jan-Ming Ho. Discovering informative content blocks from web documents. In Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '02, pages 588-593, New York, NY, USA, 2002. ACM.

[5] Ziv Bar-Yossef and Sridhar Rajagopalan. Template detection via data mining and its applications. In Proceedings of the 11th International Conference on World Wide Web, WWW '02, pages 580{591, New York, NY, USA, 2002. ACM.

[6] Lan Yi and Bing Liu. Web page cleaning for web mining through feature weighting. In Proceedings of the 18th International Joint Conference on Artificial Intelligence, IJCAI'03, pages 43{48, San Francisco, CA, USA, 2003. Morgan Kaufmann Publishers Inc.

[7] Ashraf M. Kibriya, Eibe Frank, Bernhard Pfahringer, and Geoffrey Holmes. Multinomial naive bayes for text categorization revisited. In Proceedings of the 17th Australian Joint Conference on Advances in Artificial Intelligence, AI'04, pages 488{499, Berlin, Heidelberg, 2004. Springer-Verlag.

[8] Html tag tree. http://www.openbookproject.net/tutorials/getdown/css/lesson4.html.

[9] List of main html tags. Online; accessed 25-march-2014. http://www.webseomasters.com/forum/index.php?showtopic=81.

[10] Html tag priority table. Online; accessed 25-march-2014. http://wiki.showitfast.com/SEO.

AUTHORS

Rasel Kabir was born in Dhaka, Bangladesh in 1989. He received the B.Sc (2010) & M.Sc (2012) degree in Computer Science & Engineering from University of Dhaka, Dhaka. His research interests include data mining and search engine optimization. Currently, he is working at software company as a software engineer.

Shaily Kabir received B.Sc (Hons.) and M.S in Computer Science and Engineering from University of Dhaka, Bangladesh and also M.Comp.Sc degree in Computer Science from Concordia University, Canada. Currently she is working as an Associate Professor in the Department of Computer Science and Engineering, University of Dhaka, Bangladesh. Her research interests include computer networks and network security, data and web mining, database management systems.

Shamiul Amin was born in Dhaka, Bangladesh in 1989. He received the B.Sc (2010) & M.Sc (2012) degree in Computer Science & Engineering from University of Dhaka, Dhaka. His research interests include data mining and search engine ranking technique. Currently, he is Asst. Professor, Dept. Of CSE, Dhaka International University.