Jestr

Journal of Engineering Science and Technology Review 13 (4) (2020) 173 - 187

JOURNAL OF
Engineering Science and Technology Review

Research Article

www.jestr.org

Extraction of Core Web Content from Web Pages using Noise Elimination

A. Saravanan[1,*] and S. Sathya Bama[2]

[1]*School of Computing Science, Sree Saraswathi Thvagaraia College, Tamil Nadu, 642205, India*
[2]*Coimbatore, Tamil Nadu, 642205, India*

Received 12 June 2020; Accepted 21 August 2020

Abstract

Due to the emergent of technological development, Web has evolved as the most powerful digital weapon for mankind in recent days. As the size of the web is increasing rapidly, extracting the interesting content from the web become the supreme challenge. In the meantime, the retrieved web pages have many uninteresting content blocks that are not useful for the user which also degrades the performance of content extraction. These uninteresting blocks include advertisements, banners, copyrights, navigation bars etc., and are normally named as web page noise. Removing these noises from the web pages is considered to be the primary task in pre-processing. This paper presents an approach that eliminates the noise and near duplicates for extracting significant content from the web page. The proposed method has three steps. Initially, the web page is divided into various blocks and the block which is considered as noise is removed using tag analysis and Document Object Model Tree. Secondly, the elimination of redundant blocks is carried out by computing fingerprints using modified simhash algorithm with proximity measure. From the distinct blocks, several parameters such as Titlewords, Linkwords and Contentwords are extracted. Thus, the extraction of significant content is carried out by computing the scores for each block using a weighted block scoring mechanism. The blocks having higher score values are extracted and finally, the core content is extracted from the web page. The experimental analysis has been performed and the results show that the proposed method eliminates noise in an efficient way.

Keywords: Noise Removal, Near Duplicates Removal, Tag analysis, Modified Simhash Algorithm.

1. Introduction

In recent days, Web has been transformed into a new manifestation or incarnation as the usage is more and more in all the fields of work. The web is storing various heterogeneous content such as text, image, audio and video etc. Extracting the required and useful content from the web is always a challenging task as it contains several noises. In reality, a web page is having 50% of the noise and 50% of useful core content. Web content mining aims at removing the web page noises and extracting the core content which is highly helpful for many information retrieval and topic summarization based web applications. Thus, it becomes a prominent research field to improve the performance.

When a web page is accessed from the web, it has core informative content among several noises that distracts the attention of the user form the main core content they intended to see. In specific, the non-informative contents named as noises, present in the web page include advertisements, unwanted images, banners, copyrights, navigation bars etc. Also, another disadvantage of web page noises is that bandwidth wastage. Sometimes we have experienced that when the bandwidth is low, the core content alone is displayed to the users [1]. Mayajohn and Jayasudha [2] classified the noises into various types. The details are given in Fig. 1. Web content mining aims at eliminating the noises present in the web pages and web documents.

As data mining techniques are highly helpful in mining web data, they cannot be used directly. The data mining techniques have to be modified to make them suitable for web data. This is because of various distinct characteristics of the web such as huge size, heterogeneous and dynamic nature [3]. Though web content mining seeming to be simple, it deals with several fields of research such as text mining, data mining, information retrieval and even statistics [4]. The process of eliminating the noises present in the web pages are specified as web content outlier mining [5-6]. Also, identifying the valuable information from the document with non-informative content is the most significant process as it helps the user to access the web pages in the handheld devices such as smartphones and PDAs [7-8].

This paper presents the novel method to remove the noises in the web page and extracts the significant content. The method initially divides the web page into blocks. The primary noises such as advertisements, banners and navigation bars are eliminated by analysing the HTML tags and constructed DOM tree. The secondary noises such as duplicates and near duplicates are eliminated by generating the fingerprints using the proposed modified simhash algorithm that uses frequencies and proximity of the terms in computing the weights. Finally, the significant content is extracted using the weighted block score mechanism. The experimental and performance analysis performed with 300 web pages shows that the proposed method of eliminating web noises provides better results than the existing methods.

*E-mail address: a.saravanan21@gmail.com
ISSN: 1791-2377
doi:10.25103/jestr.134.17

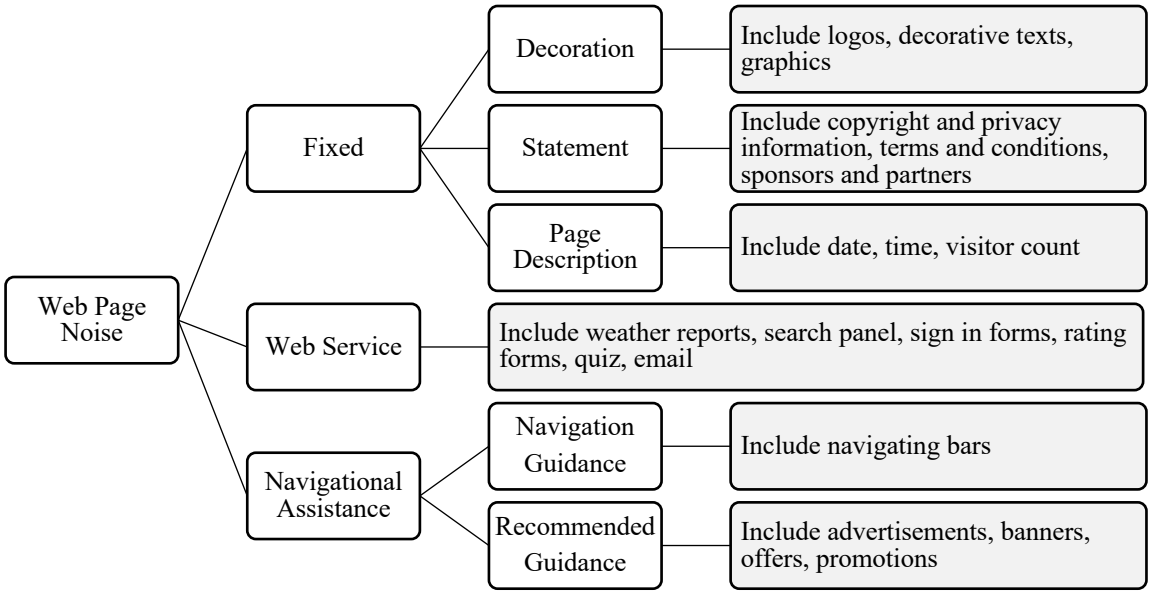

Fig. 1. Web Page Noise Classification

2. Related Work

Various techniques are proposed by the researchers in the field of web content mining [9]. Few techniques are available in the literature for eliminating noises such as primary noises, secondary noises and extracting the core contents from the web pages. In removing the primary noises present in the web pages, several tag based approaches are employed. These tags are analysed by applying the pattern recognition techniques to identify the noise templates [10]. Zeng et al. [11] proposed a V-wrapper approach which was a template independent and was used to identify and extract the core content from the news articles. The method constructs the visual tree from HTML source tags based on visual characteristics such as position, size, rich format, and term statistics.

Prasad and Paepcke [12] introduced a method named CoreEx which employs the tags based DOM tree construction. This method is very simple, however, produces good results in extracting core content form the online news web page. The method constructs the DOM tree where each node corresponds to the tags in HTML and it computes the texts and links in the nodes based on which the score is computed. ECON is another simple method proposed by Guo et al., [7] in which the DOM tree is constructed and analysed. The main and modest idea adopted by this method is that the number of punctuations present in the core content block will be always higher than the noise block. Another simple tag analysis based heuristics was suggested by Mayajohn and jayasudha [2]. The HTML tags and their attributes are analysed and if the tag is identified as a noise then the values are eliminated to produce the noise free web page core content.

Text similarity is another huge concept which plays an important role in detecting the duplicate documents. This field of study is significant in various applications such as plagiarism detection, information extraction, text summarization, document clustering and classification and even more [13-14]. Jaccard coefficient was proposed by Paul Jaccard, is a statistic used for comparing the similarity between the data sets. The Jaccard coefficient was applied between the texts using Prolog programming language. This method compares the similarity between sets of data using the intersection of sets over the union of sets [15-16].

Minhash belonging to Locality Sensitive Hashing (LSH) scheme is a powerful technique used to compute the similarity measure between the text inputs. The main idea adopted in this method is to apply k different hash functions to each pair of texts in the documents and storing the minimum value for each of the applied hash function [17-19]. Hassanian-esfahani and Kargar [20] use the minhash algorithm in detecting the near duplicates by computing the similarity between the documents. Another method that is frequently used in computing the similarity between the documents and in detecting near duplicates is simhash algorithm [21-22]. The efficiency of simhash algorithm in computing the similarity measure between the documents by generating the fingerprint is analysed and the method effectively detects the exact duplicates and near duplicates [23-24]. Hybridhash is another similarity computation approach that works similar to Simhash algorithm however simhash works for real values of data but the Hybrid hash can work on binary vectors to find the similarity. This method is highly helpful when binary data is more superfluous [25].

P.Sivakumar [24] proposed a framework to extract the relevant content by eliminating the noises present in the web page. The noises are removed using simhash algorithm and for each block, three parameters namely keyword redundancy (KR), linkword percentage (LP) and titleword relevancy (TR) are computed based on which the significant blocks are extracted using sketching algorithm. Uma and Latha [25] proposed a similar framework in which the hybrid hash is used instead of simhash and enhanced sketching algorithm is used for extracting the important contents.

3. Proposed Methodology

The proposed method aims at extracting the significant content from the web page by eliminating the irrelevant and redundant content that are considered as web page noises. These noises generally deviate the user from accessing

interesting content, degrades the performance of the search engine and even affects the rank of the web pages. Thus, removing the noises present in the web pages and extracting the significant content from a web page improves the efficiency and performance of the search engine. Conversely, identifying and segregating the noise from the web page content is very difficult as in many cases the noises present in the webpage is equal or greater than the core content.

In the proposed method, the non-informative noises such as advertisements, banners, copyrights, navigation bars are referred as primary noises whereas redundant and near duplicates are referred as secondary noises. Initially, the web pages are divided in to set of blocks. The irrelevant or non-informative primary noises present in the web page are removed using tag analysis and Document Object Model (DOM) tree construction. Once the noises are removed, the fingerprint of each block is computed using the modified simhash algorithm that uses both proximity and frequency to compute the weight for the terms from which the duplicate and near duplicate blocks are identified and are removed. Finally, the score for each block is computed by extracting prominence based parameters such as Contentwords, Linkwords and Titlewords. The weights are applied to these parameters and the final score of a block is computed using proposed weighted block scoring algorithm.

The overall framework of the proposed method is depicted in Fig. 2. The steps of the proposed framework and the methods used are listed below.

1. Block separation by analysing the tags
2. Primary noise elimination using DOM tree and tag analysis
3. Secondary noise removal with fingerprints generation using the modified simhash algorithm.
4. Extraction of significant content using block scoring technique

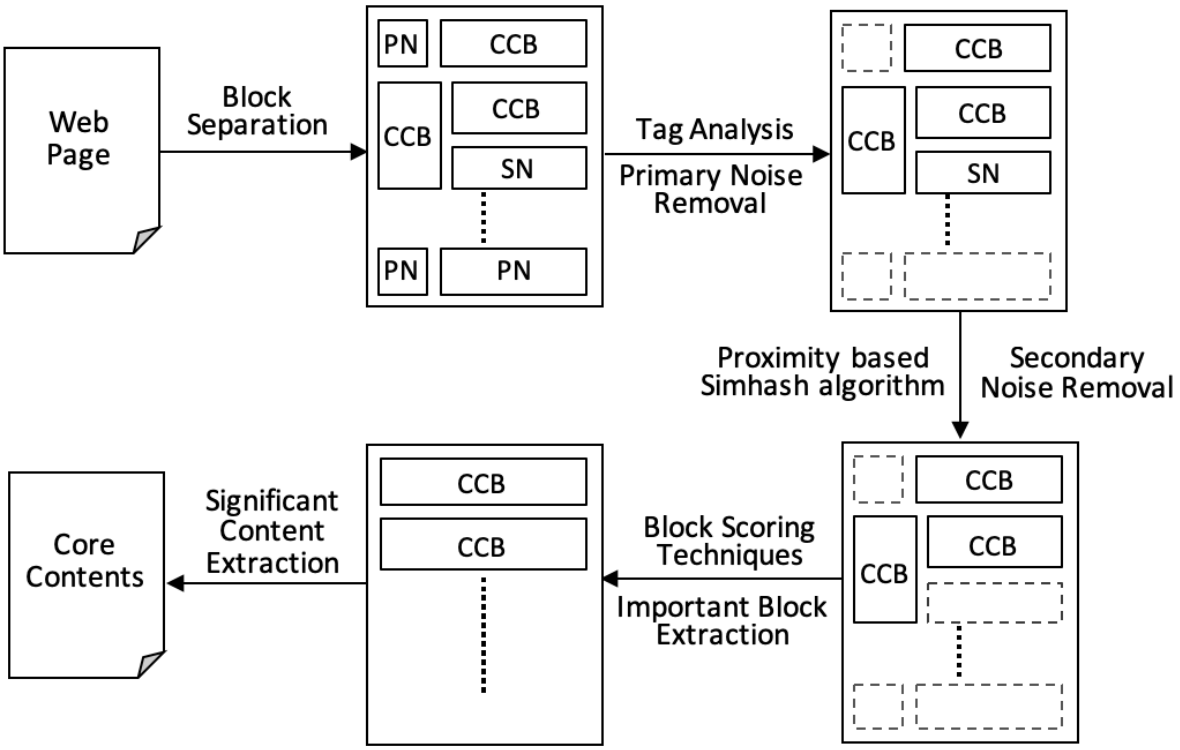

Fig. 2. Overall Framework of the Proposed Methodology

In Fig. 2, PN represents Primary Noises such as advertisements, banners, copyrights, navigation bars present in the web page, SN represents Secondary Noises such as duplicates and near duplicates present on the web page and CCB represents the Core Content Block present in the web page.

3.1 Block Separation

The user searches the web and the search engine to find the required information. The search engine crawls the World Wide Web and extracts the web pages that are relevant to the search. Finally, the resultant web pages are ranked based on the search relevancy and the raked pages are displayed to the user. However, the web pages contain not only the primary information but also the uninteresting information or primary noises that deviates the user interestingness. These uninteresting information contain several noises such as advertisements, banners, background images, plug-ins, copyright information, logos, search boxes, navigational bars, header and footer which are unnecessary for the user. Also, the presence of these noises will affect the web page rank. Though the information is useful for the website owners, it always slows down the mining process. Thus the primary noise blocks present in the web pages are to be removed for efficient mining.

Initially, the web page is divided into various smaller sections based on the structure of the content in the web page. The sections are named as blocks. The most informative core sections in the web page are Core Content Blocks (CCB). All the other uninteresting content blocks are divided into two categories such as primary or secondary. This non-content blocks having various primary noises including frames, advertisements, banners and logos, background images, header and footers, navigational bars, copyright information and so on are termed as Primary Noise Block (PNB) and content blocks containing the secondary noises such as duplicates or near duplicates are termed as Secondary Noise Block (SNB). More generally, the content of the webpage is

enclosed between various opening and closing tags. Any web page blocks are identified and separated based on the most popular HTML tags such as <BODY>, <FRAME>, <IMG>, <DIV>, <TABLE> and <TD> tags.

Any typical web page is structured in a common format where the core content is always enclosed in a <DIV> tag. This <DIV> and <TD> tags also form sections and data division in a web page. The web page is partitioned as blocks using the <DIV> and <TD> tags. The main advantage of separating the block is eliminating the primary noises present in the web page. Eliminating the noises present in the web page increases the mining efficiency and saves the storage space. Thus based on the <DIV> and <TD> tags present in the web page, blocks are identified. The proposed method considers <DIV> and <TD> as block tags enclosed inside the <BODY> tag for partitioning as blocks. The content enclosed in the <DIV> and </DIV> tags and <TD> and </TD> tags in the body of the HTML are considered as a block. However, there are some cases in which the specified block tags may have other tags along with the inner block tags as <DIV></DIV> and <P></P> inside the <TD></TD> tags. In such a case, inner block tags (<DIV>) as well as the outer block tag along with other tags (<TD><P></TD>) are considered as a separate block in the web page. Thus, the given web page (W) is separated into various blocks as $<B_1, B_2, \ldots, B_n>$, where n is the number of <DIV> tags and <TD> tags inside the <BODY> tag. These identified blocks are further analysed to extract the core content of the web page.

3.2 Primary Noise Removal Using Dom Tree and TAG Analysis

Once the blocks are separated from the web page, the tags are analysed to eliminate the primary noise. The idea behind tag analysis is that any noises present in the web pages have various characteristics. Background images are included in various HTML tags and removing the images will increase the performance. Advertisements are another major primary noise that occupies the maximum section of the web page and deviates the user to the advertisement sites. The type of advertisements data might be text, banner image, video, pop-up message etc which must be removed.

Thus, the tags are analysed and the rules are framed to eliminate the primary noises based on the typical characteristics of the web page [2]. The following are the rules to be followed for eliminating the noises.

- Serval HTML tags such as <TD>, <TR>, <TABLE>, <DIV>, <BODY> support the *background* attribute or *background-image* property with <STYLE> tag. The tags containing *background* attributes can be removed as their purpose is to style the web page. Also the <STYLE> tag or the *style* attribute is used with the *background-image* property having the URL of an image as its value specifically to display the image in the background of the content block for attracting the user. Thus, the tags <TD>, <TR>, <TABLE>, <DIV>, <BODY> and <STYLE> representing the colors or images in the *background* or *background-image* attribute can be removed.
- Most of the advertisements are displayed as an images in the web page. This can be easily identified by the verifying the domain of the web page. Most of the images related to the core content of the web page will be stored in the web server and thus the domain name representing the image in the SRC attribute will be same as a web page. Thus, if the domain specified in the SRC attribute of the image is different from the web page domain then the image can be removed. Also, the all the URLs in the web content can be compared with the URL of the ad providing website an if any of the URL specified in the content belongs to the popular ad providers, the tags can be removed as it indicates the presence of the advertisement. Likewise, the presence of the keywords such as Ad, Ads, adsbygoogle, banner and AdSense also indicates the advertisement in the web page which can be eliminated by removing the concerned tags.
- The dimension of the image plays another primary role in detecting the advertisements from other images. Most of the advertisement banners have the sizes as indicated by the Interactive Advertising Bureau (IAB). The standard dimensions of the web page banners for the Half banner, Full banner, vertical banner and head banner are 234X60, 468X60, 120X240 and 745X100 respectively. The detailed list about the sizes of the banners was given by Mayajohn & Jayasudha [2]. Thus, by analysing the size of the image the advertisements are eliminated.
- Many if the web page may embed plug-ins in order to provide the specific feature such as to play an audio or video to enhance the capability of the existing program. Generally, <EMBED>, <OBJECT> tags are used to embed the plugins in the web page. <AUDIO> and <VIDEO> tags are used to embed the audio and video in to the page. Additionally, <IFRAME> with SRC attribute also embeds video in the web page. Thus, The plug-ins can be removed by analysing the <EMBED>, <OBJECT>, SRC attribute in <IFRAME>, <AUDIO> and <VIDEO> tags which include audio, video, ActiveX and flash plug-ins.
- Not only the advertisement links, the web pages additionally contains several other links that are not interested to the user such as terms and conditions, publisher information, disclaimer, search bar, copyright information and links to social sites such as facebook, twitter, google, pinterest, linkedin. The blocks having such links can further be removed.

Once the major noises are removed, the DOM tree is constructed to remove the remaining missed noises. The Document Object Model (DOM) is a programming interface that creates a logical tree for any HTML or XML documents. As tags are the backbones of the HTML document, each tag and the data enclosed inside the tag is considered an object. The logical tree is constructed in such a way that each object is a node in the DOM tree and are organized in a hierarchical way based on the HTML page structure. The main advantage of creating the DOM tree is that it is easy to process and the object can be accessed by traversing the nodes forward and backward in the tree. Many of the browsers create a DOM tree for the web page to display. The HTML parser parses the web page and the DOM tree is constructed based on the HTML tags. Then tree is analysed to identify the minimum subtree that covers the entire web page content. The sample web page and the corresponding DOM tree is shown in Fig. 3.

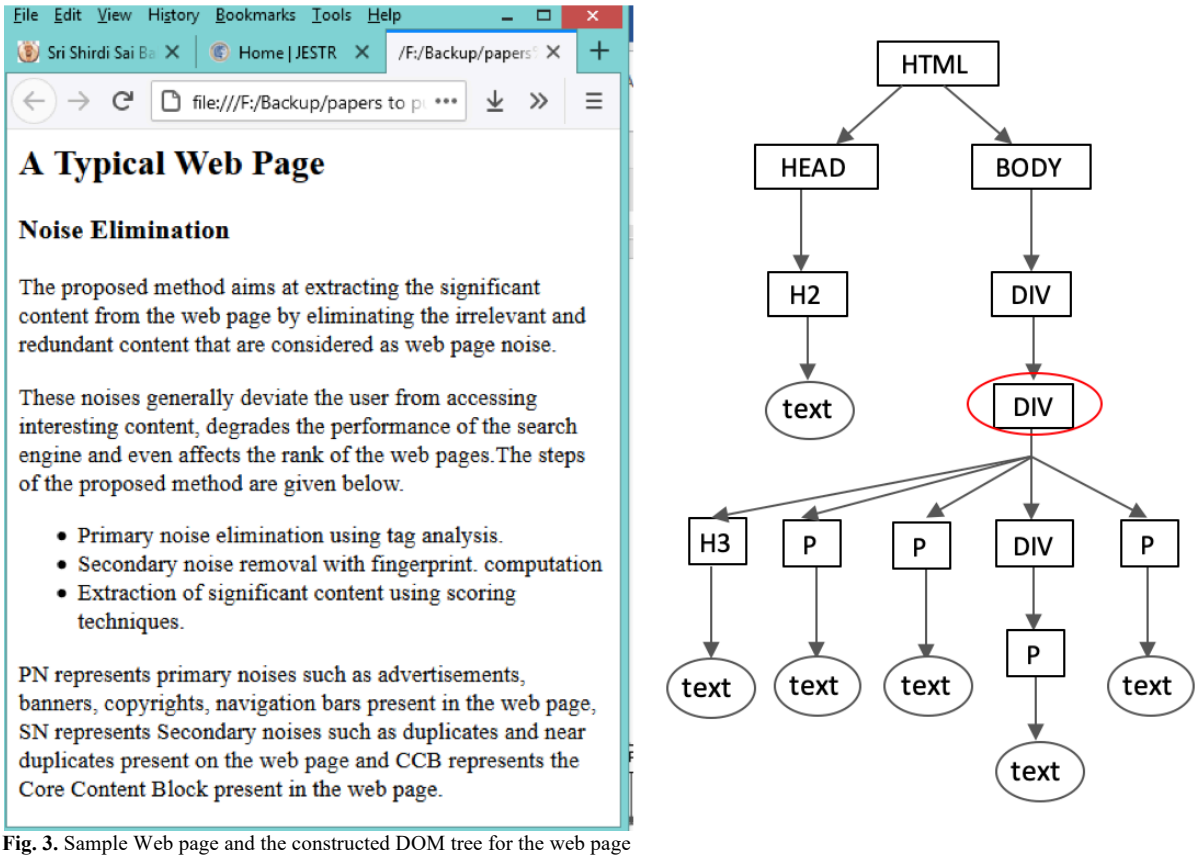

Fig. 3. Sample Web page and the constructed DOM tree for the web page

The minimum subtree of the web page is identified with a rule that a sub tree must contain the maximum content of the web page and none of its subtree wraps the entire content of the web page. From the figure, the DIV tag circled with red color is the minimum subtree that covers the maximum content of the web page. The minimum subtree is further processed to detect further noises present in the web page. The text in the descendent nodes of the DIV tag is analysed to identify the noises. The idea is to analyse punctuations present in the web content. Though it seems to be simple, the literature shows that the method provides efficient results [7]. The core content in the web page has more punctuations such as comma, periods than other noise contents. This method helps in identifying the noise from the core content.

Based on the above idea, proposed method identifies the noise by counting the number of punctuations present in the wrapped text of separated blocks. This step is carried out after block separation using <DIV> and <TD> tags and removing the noises by analysing the tags. However, some blocks seem to wrap the core content without any punctuations. Thus, to avoid false result, the size of the text is analysed. If the size of the text is small, then the text wrapped by the node is merged with the <DIV> or <TD> node adjacent to each other and then the number of punctuations is calculated. Thus, the <DIV> or <TD> tags having no punctuations are identified as the noises and are eliminated from the web page. From the DOM tree presented in Fig. 3, punctuation counts of the text wrapped inside the two <DIV> tags one with <H3>, <P>, <P>, <P> and other with <P> are computed. Finally, the blocks having no punctuation marks are removed. Thus the primary noises present in the web page is analysed based on the HTML tags in the web pages and by analysing the punctuations present in the blocks using DOM tree construction.

3.3 Duplicates and Near Duplicates Removal using Modified Simhash Techniques

As the primary noises are removed using DOM tree and tag analysis, the secondary noises present in the web pages are also to be removed. These secondary noises include both the duplicate and near duplicate blocks. Many of the methods of text mining used for identifying the duplicates and near duplicates makes use of the frequency of the terms in the document. However, the closeness of the terms in the web content plays a much more important role in computing the similarity [26]. Proximity measures the closeness of the terms in the text block. The underlying assumption of this proposed proximity based similarity measure is that the terms that appear closer together in the blocks are more comparable as it provides a higher degree of relevance. The simhash function is modified in such a way that it includes the proximity based frequency score as weight in computing the fingerprint of the blocks. The fingerprints are computed as binary numbers and are used in identifying the duplicates and near duplicates. The proposed model of the modified simhash algorithm in removing duplicates and near duplicates is shown in Fig. 4.

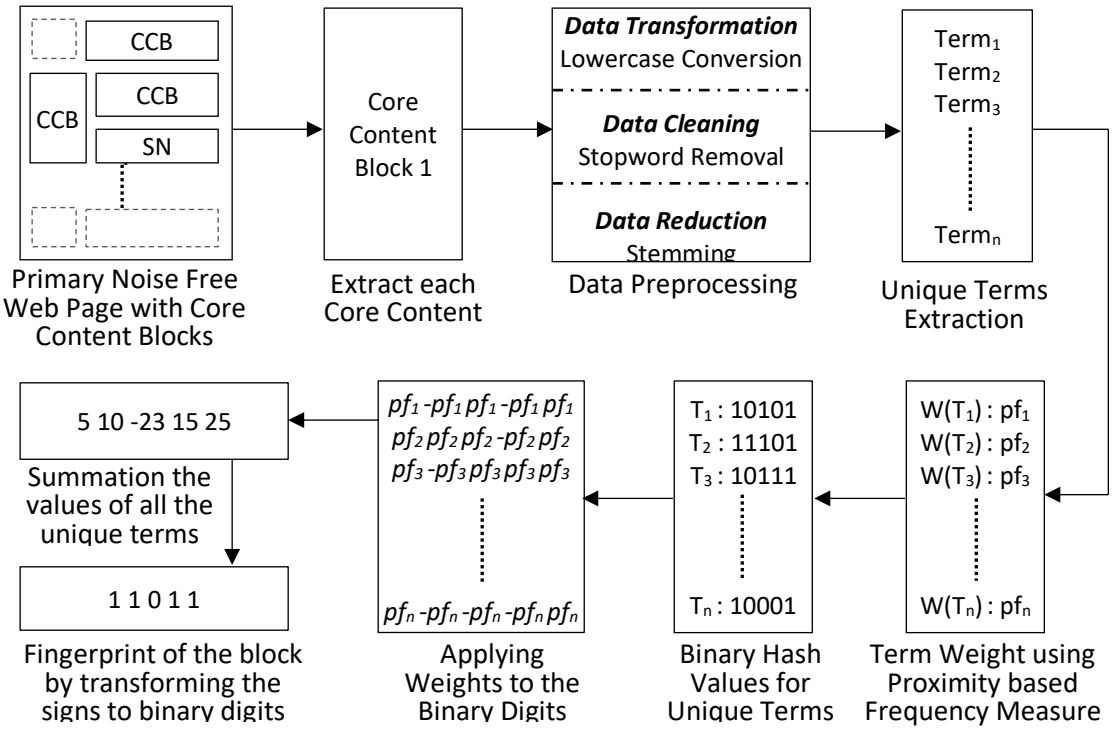

Fig. 4. Duplicates and Near Duplicates Removal using Modified Simhash Technique

The primary noise free web page blocks are given as an input for comparing the blocks to find the similarity between them. Each core content block is processed to compute its fingerprint which is used later to identify the duplicates as well as near duplicates based on the similarity in the fingerprint of the blocks. Initially, the blocks are to be preprocessed in which the core content blocks are reformed to unstructured format by eliminating all the tags in the documents. Likewise, all the typescripts in the documents are converted to lower case letters. These core content blocks are then given as an input for data cleaning in which the uninteresting stop words are removed and data reduction is applied where the significant words are reduced to its root form. The output of the preprocessing step is a set of significant terms. Thus, these steps are primary and significant on which the proposed methodologies have been implemented [27]. The weights for the unique words extracted from the blocks after preprocessing are computed based on the frequency of occurrences of the terms in that block and the proximity of the terms with the other terms. The hash function is applied to the terms which produces the fixed length binary digits. The calculated weights are allocated to each digit based on its value and finally, the digits of all the terms are calculated to produce the fingerprint of the block. Finally, all the fingerprints are compared for measuring the level of similarity to identify the duplicate and near duplicate blocks. The details about the pre-processing, modified simhash algorithm and the weight calculation are explained in the below sub-sections.

3.3.1 Data Cleaning and Data Reduction

Data cleaning in text mining eliminates the features that convey little or no meaning at all. The proposed method eliminates the audio, video and stop words which are uninteresting for the mining process. The stop words are recurrently used terms which convey a little meaning and can be eliminated in order to reduce the space and time effectively. Stop words are common words in English which carry less significant meaning than keywords, such as determiners that signify nouns like *a, an, the*; connectors that connect sentences like *and, but*; interjection that carry emotions like *Hey!, Oh!*; prepositions that act as a linker like *about, from, to*; pronouns that act as an auxiliary noun like *he, she, it*; auxiliary verbs like *is, was, have*. Thus removing these words improves the effectiveness of the mining, as these stop words have very frequent occurrences in the language than the core keywords.

Another preprocessing step is data reduction. It is the process of reducing the size of the underlying data but producing the same or similar analytical results. Data reduction in text mining can be carried out by the process called stemming. Stemming is the practice of shrinking the transformed or derived words to their root form or base form. It removes word suffixes to diminish the quantity of unique words, thereby, decreasing the storage space and speeds up the search process. For instance, words such as performance, performer, performing, performable, performs are related and derived from the word perform. Generally, the information retrieval system accomplishes stemming for enlightening the performance and increasing the search speed. Several algorithms are available to perform the stemming process. However, the proposed method uses the porter stemming algorithm to perform stemming which is a process of data reduction.

3.3.2 Term Weight Calculation using Proximity based Frequency Measure (TWC-PFM)

Once the identical terms are identified, the weights are computed for each unique terms in the core content blocks. This weight computation makes use of both frequency of occurrence and the closeness of the term with respect to the other terms. The proximity of the terms with other terms is computed by assigning position values to the terms extracted. The working procedure is explained with an example.

Consider a core content block CCB_1 in a web page W. Let the terms in the CCB_1 be {a b c d a b a c a b d}. The set of

unique terms {T_1, T_2, …, T_n} be {a, b, c, d} in which n is 4. Assigning the integer value that represents the position of the content will result in P = {1 2 3 4 5 6 7 8 9 10 11}. To calculate the weight of all the unique terms based on the proximity and frequency of the terms extracted from CCB_1, the position vector for each unique term a, b, c, d in the CCB_1 is extracted as P(a) = {1, 5, 7, 9}, P(b) = {2, 6, 10}, P(c) = {3, 8} and P(d) = {4, 11} respectively. Also the frequency of all the extracted unique terms a, b, c and d in CCB_1 is f(a) = 4, f(b) = 3, f(c) = 2 and f(d) = 2.

Based on the concept of the proximity or closeness of the terms, there are two factors which affect the computed score. The first parameter is the distance between the term-pairs in the content and the second parameter is the order of occurrence of the terms in term-pairs. Thus the distance between all the adjacent term pair t_i and t_j is computed which preserves the order of occurrences of the term-pair based on the identified position vectors of the terms in the content block. Thus, for the above illustration, the adjacent term-pairs are engendered as ab, ac, ad, bc, bd, ba, ca, cb, cd, da, db and dc. The distance between any term-pair can be computed as given in Eq. 1.

$$dis(t_i, t_j) = P(t_i) - P(t_j) \quad (1)$$

where dis(t_i, t_j) is the distance between two unique terms t_i and t_j and P(t_i) and P(t_j) is the position of the unique terms t_i and t_j respectively. However, the basic constraint in computing the proximity value between the terms in a document is that the value should decrease as the distance between the terms increases. This implies that the proximity and the value are inversely proportional to each other. Thus the terms that appear very close to each other frequently will have a higher score and on the other hand, the terms that are far away from each other are not much related to each other and therefore they have a lower score. Thus the Term Proximity TP(t_i, t_j) can be computed as given in Eq. 2.

$$TP(t_i, t_j) = \frac{1}{dis(t_i, t_j)} \quad (2)$$

When more than one t_j occurs near t_i or more than one t_i occurs near t_j, the one with the smallest distance is taken into account for computing the proximity score. However, the distance between all the occurrences of the term pairs t_i and t_j must be computed. The distance between the term-pairs is computed by comparing the position vectors of the term individually. The positions of the two terms along with the next position are compared for computing the distance.

For comparing the position vectors P(t_i) and P(t_j), two pointers p_i and p_j can be used to point the current positions of the terms t_i and t_j respectively. If the value of p_i is less than the value of p_j and the value of p_{i+1} is greater than the value of p_j, then the term proximity of the corresponding occurrence of the terms t_i and t_j is calculated using the formula given in Eq. (1) and Eq. (2). In case if more than one t_i occurs near t_j, which means if both the value of p_i and p_{i+1} is less than the value of p_j then the pointer p_i is incremented by one. However, if more than one t_j occurs near t_i, which means if the value of p_i is greater than the value of p_j then the pointer p_j is incremented by one. The process continues until all the positions in the vectors P(t_i) or P(t_j) is processed completely.

Thus the weight for each term t_i is computed by summing the frequency of the term along with the sum of all the possible term pairs with respect to the term t_i. Thus the formula to compute the final weight of the term is given in Eq. 3.

$$w(t_i) = \frac{f(t_i) + \sum_{j=1}^{l} \sum_{k=1}^{m} TP(t_i, t_j)}{N_t} \quad (3)$$

Where m is the number of occurrences of the same term-pairs and l is the number of term-pairs with respect to t_i. f(t_i) is the number of occurrences of the term t_i. N_t act as a scaling factor which represents the total number of unique terms in the web page core content block. Thus the weight for all unique terms in the document is computed based on the frequency and the proximity of the corresponding term-pairs. The pseudocode for calculating the weight of the terms is given below.

Algorithm 1: Pseudocode for Term Weight Calculation
Input: Web Page W
Output: Weights of all the terms
Method: Term Weight Calculation using Proximity based Frequency Measure
Function TWC_PFM(T)
Begin
Unique Terms UT = {}; N_t = 0
For the terms t_i from T
If the term t_i is in the UT set
Add the position to the t_i's position vector P(t_i)
//Increment the frequency of the term by 1
Frequency(t_i) = Frequency(t_i) + 1
Else
Add the term t_i in the UT set
$N_t = N_t + 1$
Create a position vector P(t_i) and add the position into it
//Initialize the frequency of the term to 1 (first occurrence)
Frequency(t_i) = 1
End If
Generate all possible term-pairs TP
End For
For each term t_i in the UT
w(t_i) = Frequency(t_i)
For each term-pairs (t_i, t_j)
While (P(t_i) or P(t_j) = EOF)
Assign pointers ptr_i & ptr_j to the first element in the position vectors P(t_i) & P(t_j)
If (($*ptr_i < *ptr_j$) and ($*ptr_{i+1} > *ptr_j$)) then
$w(t_i) = w(t_i) + (\frac{1}{*ptr_j - *ptr_i})$
Else if (($*ptr_i < *ptr_j$) and ($*ptr_{i+1} < *ptr_j$)) then
$ptr_i = ptr_{i+1}$
Else if (($*ptr_i > *ptr_j$) and ($*ptr_{i+1} > *ptr_j$)) then
$ptr_j = ptr_{j+1}$
End IF
End For
$w(t_i) = \frac{w(t_i)}{N_t}$
End For
Return Weight of all the terms W
End Function

These computed weights and the unique terms are given as an input for the modified simhash function which is explained in the next sub-section.

3.3.3 Modified simhash Algorithm

After computing the weights for each unique term in the core content blocks, the next step is to generate the fingerprint which is the unique identification of the block having fixed length value. This is done by applying the modified simhash algorithm which makes use of the weights computed using the algorithm given in Fig. 5. The fingerprint of the block is considered to be a fixed length unique value representing the block based on its content. In mathematics, hash functions are considered to be the powerful tool that maps the data to the fixed length value [28]. For computing the similarity between the blocks or documents, simhash algorithm was proposed by Charikar [29] which is a special hash function and efficient locality sensitive hashing (LSH) variant. This algorithm works only for the text documents and uses a frequency as a weight for computing the fingerprint of the text content. The main advantage of the simhash algorithm is that it is fast and space efficient. However, the algorithm doesn't consider the proximity of the term-pairs present in the document for weight computation. The algorithm has been modified to use both the frequency and the proximity for computing the weights of the terms. The working procedure of the modified simhash algorithm is depicted in Fig. 5.

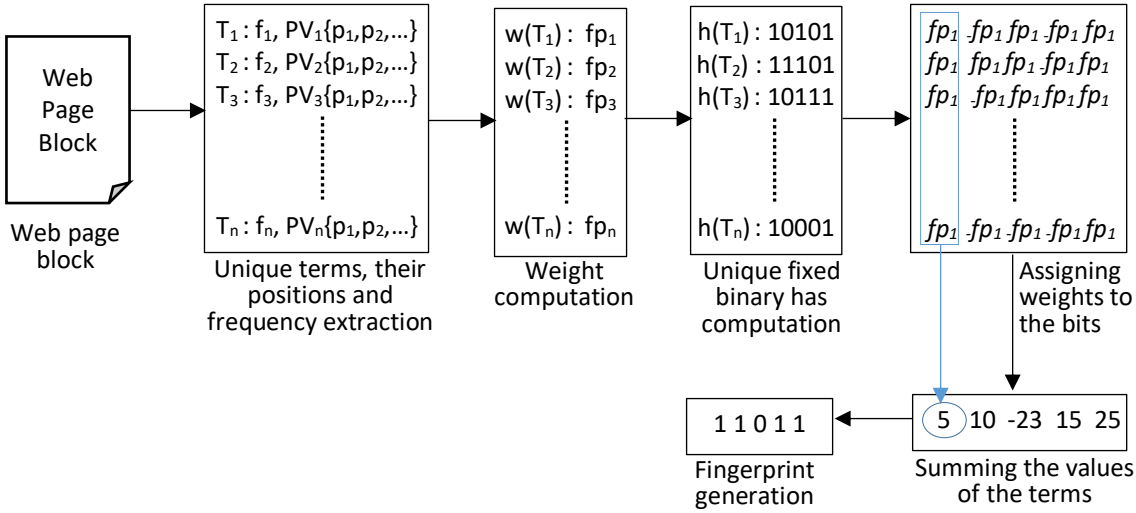

Fig. 5. Working Procedure of the Modified simhash Algorithm

The algorithm takes the given text document or web page block as an input for which the fingerprint has to be generated. The document undergoes the preprocessing steps to eliminate the stop words and to reduce the words to its root form. Each term in the document is processed to identify the unique key terms. The position vectors having the positions of the term and the frequency of occurrence of the term are also identified. Based on the frequency and the positions of the terms, the weight is computed for the terms based on the given algorithm.

The hash value is computed for all the unique terms. The hash algorithm used in the work is xxHash. xxHash is a tremendously fast non-cryptographic hash algorithm. The speed of the algorithm is close to RAM limits. The algorithm has two variations which produce the output as 32 bits and 64-bits. The 64-bit xxHash algorithm is employed in the work in which the hash algorithm takes the string of any length as input and produces the 64-bit binary code as output. This algorithm is used mainly to check the integrity of the message as the algorithm produces the same binary code for the same input string. The computational speed of the algorithm is 30 times faster than the MD5 algorithm. Once the hash values are computed, the weight of the terms is assigned to each bit in the hash value with positive weights representing the bit 1 and negative weights representing 0. Finally, each bit of all the terms is summed and the resulting value is converted to the block fingerprint. Each value corresponding to the bit is verified and if the final value is positive, the corresponding fingerprint bit is 1 else the bit is assigned as 0. Thus the fingerprint of the input document or the block is computed. The algorithm for the modified simhash is given below.

Input: Web Page Block B, Web Page W
Output: Fingerprint of the Block B
Method: Modified Simhash Algorithm for Block Fingerprint Generation

```
Function Modified_Simhash(WebPageBlock B)
Begin
    //Initialize the bits of the fingerprint vector to 0
    Int fingerprint[0..(f-1)] = 0;
    For each unique term T in WebPage B
            T is hashed into an f-bit hash value H;
            Weight(T) = TWC_PFM(W)
            For (i=0; i<h; i++) // For each bit h of the hash
value H
                If X[i]==1 Then
                    fingerprint [i]= fingerprint [i]+weight(T);
                Else
                    fingerprint [i]= fingerprint [i]-weight(T);
                End If
            End For
    End For
    For (i=0; i<h; i++)
            If (fingerprint[i]>0) Then
                fingerprint[i] = 1
            Else
                fingerprint[i] = 0
            End If
    End For
End Function
```

The modified simhash algorithm is applied to all the retrieved web page block to generate their fingerprints respectively. These fingerprints are compared to detect the

duplicates and near duplicate blocks. For labeling the blocks as near duplicates, another parameter k is used where the fingerprints of the two blocks that differ atmost k bits is considered to be the near duplicate and can be eliminated. Additional care must be taken in choosing the value for the parameter k, as very low values has a chance of missing the duplicates and near duplicates and very high values consider the large number of blocks including distinct block as duplicates. From the literature, choosing the value for k as 3 is considered to be reasonable as it increases the accuracy, precision and recall values. With 64-bit fingerprints, two documents are considered as near-duplicates if their fingerprints differ by at most 3 bits [30]. Thus, the method eliminates the secondary noise such as duplicates and near duplicates blocks and presents only the distinct blocks present in a web page.

3.4 Weighted Block Scoring Algorithm

The distinct blocks extracted by applying the sequential operations such as DOM tree with tag analysis and the modified simhash algorithm that aims at eliminating the primary and secondary noises present in the web page is served as an input for the block scoring algorithm. The score for each extracted distinct block is computed from which the important blocks are extracted. The formula to compute the weighted block scoring mechanism is given in Eq. 4.

$$Block_{Score}(i) = 1 - \left(\alpha * T_w(i) + \beta * L_w(i) + \gamma * C_w(i)\right) \quad (4)$$

$Block_{Score}(i)$ represents the score of the block i that lies between 0 and 1 computed using three parameters such as $T_w(i)$, $L_w(i)$ and $C_w(i)$. $T_w(i)$ represents the relevance weightage of the Titlewords in the block i, $L_w(i)$ represents the relevance weightage of the Linkwords present in the block i, $C_w(i)$ represents the relevance weightage of the Contentwords in the block i. n represents the content terms in the block i. α, β and γ are the weight factor with two constrains such as $\alpha \leq \beta \leq \gamma$ and $\alpha + \beta + \gamma = 1$. The values of the weights are $\alpha = 0.3$, $\beta = 0.3$ and $\gamma = 0.4$. Each parameter is explained below.

Relevance Weightage of the Titlewords

This parameter provides the weightage to the Titlewords present in the block. $T_w(i)$ is the weightage computed for the Titlewords present in the block *i*. The formula is given in Eq. 5.

$$T_w(i) = \left(\frac{title\_count_i}{N_{Distinct\_title}}\right) \quad (5)$$

where $title\_count_i$ represents the number of Titlewords present in the block *i*. $N_{Distinct\_title}$ is the number of distinct title terms in the web page.

Relevance Weightage of the Linkwords

This parameter provides the weightage to the Linkwords present in the block. $L_w(i)$ is the weightage computed for the Linkwords present in the block *i*. The formula is given in Eq. 6.

$$L_w(i) = \left(\frac{link\_count_i}{N_{Distinct\_link}}\right) \quad (6)$$

where $link\_count_i$ represents the number of Linkwords present in the block *i*, $N_{Distinct\_link}$ is the number of distinct link terms in the web page.

Relevance Weightage of the Contentwords

This parameter provides the weightage to the Contentwords present in the block. $C_w(i)$ is the weightage computed for the Contentwords present in the block *i*. The formula is given in Eq. 7.

$$T_w(i) = \left(\frac{content\_count_i}{N_{Distinct\_content}}\right) \quad (7)$$

where $conetnt\_count_i$ represents the number of Contentwords present in the block *i*, This can be computed by subtracting the title_count and link_count from the number of terms N in the block *i*. $N_{Distinct\_content}$ is the number of distinct Contentwords present in the webpage.

Thus, the scores for the distinct web page blocks are computed to extract the significant blocks that better servers the user that comply with their interestingness. Once the scores are extracted a threshold level has to be set for the scores to differentiate the significant and irrelevant content. Thus the threshold is set by computing the ratio between the number of distinct blocks extracted from the web page to the number of blocks in the web page. The threshold value is computed using Eq. 8.

$$t_{scoring} = \frac{Number\ of\ distinct\ blocks}{Total\ Number\ of\ blocks} \quad (8)$$

The algorithm for extracting the significant blocks from the web page is given below.

Input: Set of n Distinct Web Page Blocks B, Total Number of Web page blocks m
Output: Significant Block Extraction
Method: Weighted Block Score Algorithm
Function BlockScore (WebPageBlocks B)
Begin
//Initialize the score for the set of web page blocks to 0
Threshold t = n/m
Read the Web page once and compute
distinct number of title word d_tw,
distinct number of link word d_lw
distinct number of content word d_cw
For each block b in the set B
Initialize Titleword, Linkword and Contentword to 0
For each term t in the block
If term t is in the title
Titleword = Titleword + 1
Else if term t is a link
Linkword = Linkword + 1
Else
Contentword = Contentword + 1
End IF
End For
w_t = (Titlecount/d_tw) //weightage for title terms in a block
w_l = (Linkcount/d_lw) //weightage for link terms in a block
w_c = (Contentword/(d_cw)) //weightage for content terms in a block

$$\text{Blockscore}(b) = 1 - (0.3 * w\_t + 0.3 * w\_l + 0.4 * w\_c)$$

If Blockscore(b) > t then
Significant Blocks = Significant Blocks ∪ b
End For
Return Significant Blocks
End Function

The output of this step is the significant blocks extracted from the web page after eliminating the primary noises such as advertisements, banners, etc., and the secondary noises such as duplicates and near duplicates. Thus the proposed mining method can be used to extract the significant core contents from the web page by eliminating the noises present in it.

4. Experimental Analysis

The experimental analysis has been performed for the proposed method at each stage and the results are discussed in this section. Generally, many of the web pages consist of primary and secondary noises which deviate the users' interestingness towards other than the intended one. The primary noises can be eliminated by analysing the tags of the particular web page. For example, the source code of the sample webpage available at the URL "https://www.tutorialride.com/data-mining/web-mining.htm" is shown in Fig. 6. On the left-hand side of the Fig. 6, the source code of the web page contains uninteresting links to social sites such as facebook, twitter, googleplus and linkedin with the link which is different from the web page domain. Similarly, on the right-hand side of the Fig. 6, the presence of the keyword *adsbygoogle* indicates the advertisement in the web page. Thus these are primary noises that are present in most webpages which must be removed for efficient web content mining.

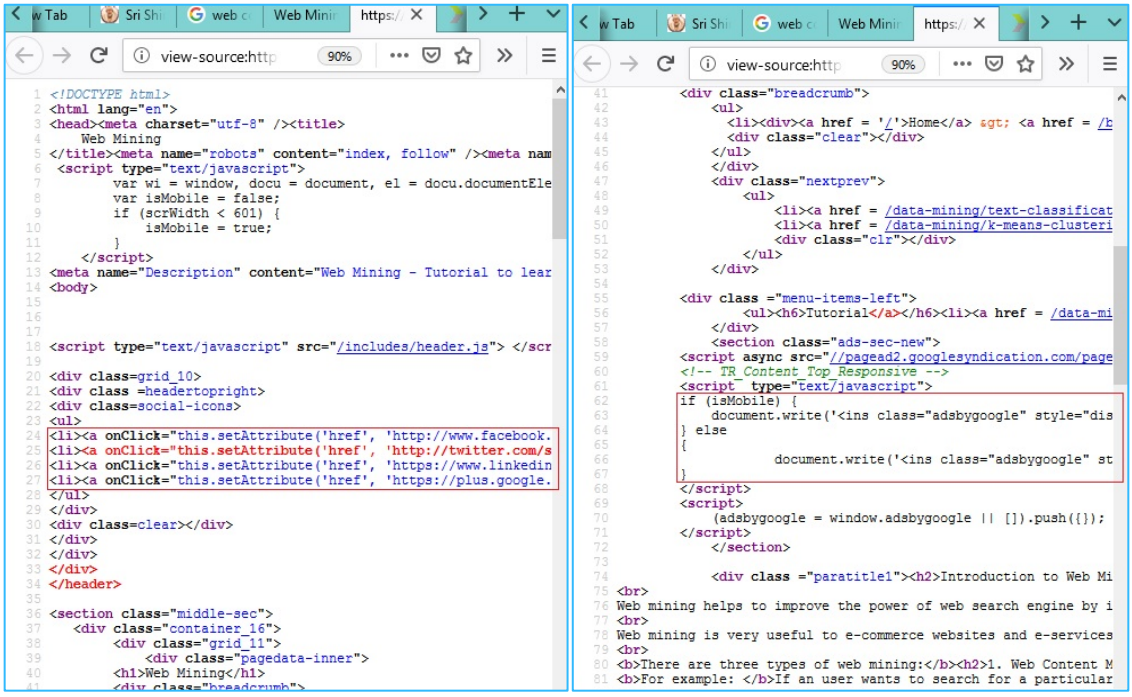

Fig. 6. Sample Web Page for Tag Analysis

The web page used for the analysis of the proposed work is available at "https://www.tutorialspoint.com/data_mining/dm_evaluation.htm". This web page termed as wp, undergoes a detailed analysis on the HTML tags. The web page is initially divided into 14 blocks and the blocks are identified using block id b1, b2, …, b14. The blocks are divided based on the <DIV> tag. The rules presented in section 3.2 is analysed on the web page blocks for eliminating web page blocks that constitute the primary noise. The HTML tags are analysed for the uninteresting links to social sites and the links present in the web page such as facebook, googleplus, twitter, linkedin and youtube and are removed. The advertisement banners of size 468X60 in the web page is also removed. The <DIV> tags with class name *bottomgooglead, rightgooglead and simple-ad* indicating google advertisements as well as the clock has been removed as it is the primary noise block. After removing the primary noise blocks, the web page has been reduced to 9 blocks. DOM tree is constructed for the resultant blocks based on the HTML tags. The tree is analysed to identify the minimum subtree that covers the entire web page content by computing the punctuation marks. Thus after applying this process, the web page has been ended with 8 blocks such as b1, b4, b5, b9, b10 and b14. The details such as block ID, the method used in identifying the content details of the primary web page blocks that can be removed are given Tab. 1.

Table 1. Primary Noise Block identified from the webpage

S.No	Block ID	Method used	Details of the Content
1	b1	Tag Analysis	Social Links such as facebook, googleplus, twitter, linkedin and youtube
2	b4	DOM Tree Analysis	Uninteresting block with the heading Selected Reading and identified using DOM tree analysis
3	b5	Tag Analysis	Advertisement image of size 468x60
4	b9	Tag Analysis	Gogole Advertisement at the bottom of the page

5	b10	Tag Analysis	Gogole Advertisement at the Top of the page
6	b14	Tag Analysis	Footer details such as privacy policy and contact us links

After eliminating the primary noises identified using DOM tree and tag analysis, the resultant web page contains the blocks such as b2, b3, b6, b7, b8, b11, b12 and b13. These web blocks are further processed to identify the presence of secondary noises. The contents present in the blocks are pre-processed by transforming the words to unstructured format by removing tags, removing the stop words present in the content and reducing the terms to its root form using porter stemming algorithm. The weights for the distinct terms present in each web page block are computed using proximity based frequency measure. The hash values for the distinct terms are computed using 64 bit xxHash algorithm. The modified simhash algorithm is applied to generate the fingerprint of the web page blocks. The binary fingerprint of the blocks computed using the modified simhash algorithm is given in Tab. 2.

Table 2. 64-bit Binary Fingerprint generated for the Web Page Blocks

S.No	Block ID	64-bit Binary Fingerprint
1	b2	10110101 00011101 11000001 10101100 11011011 10101110 11000011 00000111
2	b3	10111010 10010101 11001100 10011001 11100010 10110101 00101010 11011011
3	b6	10100011 10010110 00110011 01101010 01101101 01010011 00100011 01010011
4	b7	10110101 **1**0011101 11000001 10101100 11011011 1**00**01110 11000011 **1**0000111
5	b8	10010100 00110100 10101101 11001000 10111100 10101010 00101110 11110010
6	b11	10010110 00110001 10111101 11000001 10101101 10001011 00101010 10110010
7	b12	11001110 00011101 10101100 01010011 01001010 00101110 01010111 11011011
8	b13	11001010 01011101 10111100 01011011 01000010 00001110 01010101 10011011

For identifying duplicate blocks and near duplicate blocks, the fingerprints of the two blocks are compared and the blocks that differ with atmost 3 bits are considered as similar blocks. Based on this, one of the block is considered as the near duplicate and can be eliminated. The comparison on 64-bits is made by performing XOR operation between the fingerprints in which the number of 1's identified from the results signifies the bit difference between the fingerprints. The block pairs having the atmost difference with 3 bits can be marked as near duplicates. Thus, by comparing the fingerprints of blocks, two blocks b2 and b7 are having similar bits with the difference of 3 bits. Block b7 is labeled as the near duplicate block and is eliminated to improve efficiency. The result of the comparison process in identifying the distinct and noise block is given in Tab. 3.

Table 3. Block Comparison based on the bits in identifying duplicate blocks

S.No	Block ID	Block Comparison							Distinct / Duplicate Block
		b3	b6	b7	b8	b11	b12	b13	
1	b2	31	34	***3***	30	34	31	35	Distinct Block
2	b3	-	33	3	27	31	26	28	Distinct Block
3	b6	-	-	33	32	30	35	35	Distinct Block
4	b7	-	-	-	31	32	32	34	Duplicate Block (same as b2)
5	b8	-	-	-	-	12	29	35	Distinct Block
6	b11	-	-	-	-	-	31	33	Distinct Block
7	b12	-	-	-	-	-	-	8	Distinct Block
8	b13	-	-	-	-	-	-	-	Distinct Block

As a result, 7 blocks such as b2, b3, b6, b8, b11, b12 and b13 are extracted as the distinct blocks after the second phase of removing near duplicates. The next phase is extracting the significant blocks using block scoring mechanism explained in Section 3.4. Weighted Block Score algorithm is employed to compute the significance score for each distinct blocks. The details of the computation of the weighted block score is given in Tab. 4.

Table 4. Computation of Weighted Block Score

Block ID	Total Number of Words	Total No. of Title Words	Title Relevancy Computation	Total No. of Link Words	Link Relevancy Computation	Total No. of Content Words	Total No. of Distinct Words	Content Relevancy Computation	Block Score
b2	46	7	0.583	0	0.000	39	28	0.172	0.756
b3	40	0	0.000	4	0.444	36	25	0.159	0.803
b6	33	0	0.000	0	0.000	33	27	0.145	0.942
b8	69	7	0.583	7	0.778	55	33	0.242	0.495
b11	68	10	0.833	6	0.667	52	42	0.229	0.458
b12	36	0	0.000	0	0.000	36	36	0.159	0.937
b13	41	0	0.000	0	0.000	41	36	0.181	0.928

Total No. of Distinct Title Words in a Web Page : 12
Total No. of Distinct Link Words in a Web Page : 09
Total No. of Distinct Content Words in a Web Page : 227

The threshold value is computed using Eq. (8). The total number of blocks in the web page is 14 and the total number of distinct blocks extracted is 7. Thus the threshold value for the block score is computed as 0.5. From the analysis, the block score of the blocks b8 and b11 are having lesser score than the threshold value and are considered as irrelevant content. Thus, the blocks b2, b3, b6, b12 and b13 having the

scores greater than the threshold value, these are considered as the significant blocks.

5. Performance Analysis

The performance of the proposed system is evaluated and is compared with the existing methods. The experiments are performed on a machine with 1.70 GHz Intel Core processor and 4 GB RAM. For conducting the experiments, 300 web pages are collected from 30 news sites in English. The details about the websites are given in Tab. 5.

Table 5. News Websites used for the Experiment

S.No.	News Websites	URL of the News Site
1	Times of India	https://timesofindia.indiatimes.com/
2	New Delhi Television Limited	https://www.ndtv.com/
3	India Today	https://www.indiatoday.in/
4	The Indian Express	https://indianexpress.com/
5	The Hindu	https://www.thehindu.com/
6	CNN News18	https://www.news18.com/
7	Firstpost	https://www.firstpost.com/
8	Business Standard	https://www.business-standard.com/
9	DNA	https://www.dnaindia.com/
10	Deccan Chronicle	https://www.deccanchronicle.com/
11	Oneindia	https://www.oneindia.com/
12	The Financial Express	https://www.financialexpress.com/
13	BusinessLine	https://www.thehindubusinessline.com/
14	The Quint	https://www.thequint.com/
15	Outlook India	https://www.outlookindia.com/
16	Free Press Journal	https://www.freepressjournal.in/
17	Telangana Today	https://www.freepressjournal.in/
18	The Asian Age	https://www.asianage.com/
19	Chandigarh Metro	https://chandigarhmetro.com/
20	Daily Excelsior	http://www.dailyexcelsior.com/
21	The Navhind Times	http://www.navhindtimes.in/
22	The Arunachal Times	https://arunachaltimes.in/
23	The Sangai Express	http://www.thesangaiexpress.com/
24	State Times	http://news.statetimes.in/
25	News Today	https://newstodaynet.com/
26	Financial Chronicle	http://www.mydigitalfc.com/
27	The Times of Bengal	http://www.thetimesofbengal.com/
28	The News Himachal	https://www.thenewshimachal.com/
29	League of India	https://leagueofindia.com/
30	The Eastern Herald	https://www.easternherald.com/

Three various experiments are conducted for the three stages such as primary noise removal, secondary noise removal and significant block extraction of the proposed method. The performance analysis has been made by manually comparing the results produced by the methods. The web pages extracted by the proposed method and existing methods are manually analysed and if the web page blocks extracted for the given input web pages are core contents, then it is said to be correctly extracted. On the other hand, if the extracted webpages contain noise blocks other than the core content, then it is said to be incorrectly extracted.

Experiment 1: Performance of Primary Noise Removal

Experiment 1 is performed to evaluate the performance of primary noise removal techniques. The proposed primary noise removal using DOM tree and Tag analysis is compared with the existing methods such as V-Wrapper [11], CoreEx [12], ECON [7] and Tag based [2]. The analysis has been done by varying the number of web pages. The details such as the number of correctly extracted web ages and their percentage and the number of incorrectly extracted web pages and their percentage are given in Tab. 6.

Table 6. Performance Analysis for Primary Noise Removal

No. of Web pages	Methods	No. of Correctly Extracted Web Pages	Correctly Extracted Web Pages (in %)	No. of Incorrectly Extracted Pages	Incorrectly Extracted Pages (in %)
100	V-Wrapper	78	78.00	22	22.00
	CoreEx	81	81.00	19	19.00
	ECON	90	90.00	10	10.00
	Tag-based	92	92.00	8	8.00
	Proposed	93	93.00	7	7.00
200	V-Wrapper	164	82.00	36	18.00
	CoreEx	171	85.50	29	14.50
	ECON	182	91.00	18	9.00
	Tag-based	185	92.50	15	7.50
	Proposed	189	94.50	11	5.50
300	V-Wrapper	242	80.67	58	19.33
	CoreEx	256	85.33	44	14.67
	ECON	264	88.00	36	12.00
	Tag-based	271	90.33	29	9.67
	Proposed	272	90.67	28	9.33

The average of correctly extracted web pages at each iteration with varied number of web pages for all the

individual methods has been computed for evaluating the effectiveness of the systems. The average is computed by dividing the sum of correctly extracted web pages by the sum of varied number of web pages used under the study at each iteration. The average of correctly extracted web pages for the methods such as V-Wrapper, CoreEx, ECON, Tag-based and Proposed methods are 80.22%, 83.94%, 89.67%, 91.61% and 92.72%. Thus the proposed DOM tree and tag analysis based method provide better efficiency in removing the outliers or primary noises present in the web pages. The accuracy of the methods is compared in Fig. 7.

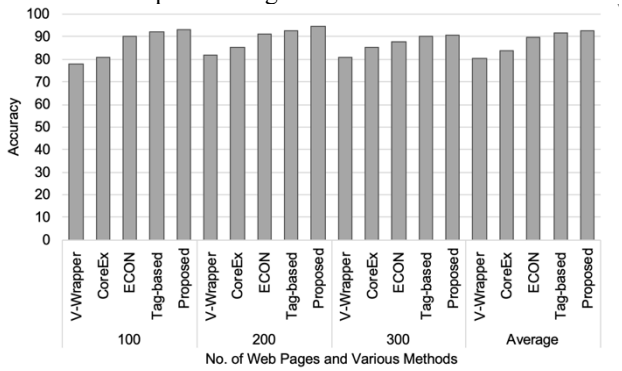

Fig. 7. Accuracy Comparison for Primary Noise Removal

Experiment 2: Performance of Secondary Noise Removal

The second experiment is performed to evaluate the performance of the method in removing the secondary noises such as duplicates and near duplicates present in the web pages. Several existing methods such as Jaccard Coefficient, Minhash, Simhash, Hybridhash exist to compute the similarity between the text documents. Thus, these existing methods and the proposed modified simhash method are executed for the set of web pages and the results produced are compared manually to compute the performance. The details such as the number of correctly extracted web ages and their percentage and the number of incorrectly extracted web pages and their percentage are given in Tab. 7.

Table 7. Performance Analysis for Secondary Noise Removal

No. of Web pages	Methods	No. of Correctly Extracted Web Pages	Correctly Extracted Web Pages (in %)	No. of Incorrectly Extracted Pages	Incorrectly Extracted Pages (in %)
100	Jaccard Coefficient	85	85.00	15	15.00
	Minhash	89	89.00	11	11.00
	Simhash	89	89.00	11	11.00
	Hybridhash	90	90.00	10	10.00
	Modified Simhash (Proposed)	91	91.00	9	9.00
200	Jaccard Coefficient	162	81.00	38	19.00
	Minhash	167	83.50	33	16.50
	Simhash	170	85.00	30	15.00
	Hybridhash	172	86.00	28	14.00
	Modified Simhash (Proposed)	179	89.50	21	10.50
300	Jaccard Coefficient	237	79.00	63	21.00
	Minhash	242	80.67	58	19.33
	Simhash	249	83.00	51	17.00
	Hybridhash	251	83.67	49	16.33
	Modified Simhash (Proposed)	253	84.33	47	15.67

For better clarification on the overall effectiveness of the systems under study, the average values are computed for all the methods by dividing the sum of correctly extracted web pages by the sum of varied number of web pages used at each iteration. The average of correctly extracted web pages for the methods such as Jaccard Coefficient, Minhash, Simhash, Hybridhash and Proposed methods are 81.67%, 84.39%, 85.67%, 86.56% and 88.28%. Thus the proposed modified simhash algorithm provides better efficiency in removing the outliers such as duplicates and near duplicates constitutes the secondary noises present in the web pages. The accuracy of the methods is compared in Fig. 8.

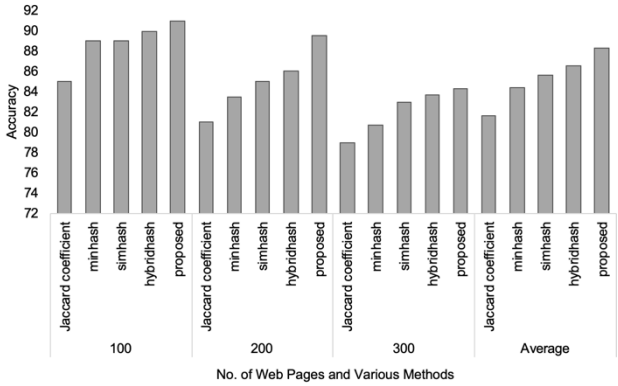

Fig. 8. Accuracy Comparison for Secondary Noise Removal

Experiment 3: Performance of Significant Core Block Extraction

The third experiment is performed to evaluate the performance of significant block extraction from the set of input web pages using modified simhash with the weighted block scoring algorithm. The existing techniques such as a method that uses simhash algorithm and sketching algorithm in extracting the relevant content form the web page proposed by the P. Sivakumar [24] termed as *Exiting 1* and a method that uses hybridhash algorithm termed as and enhanced sketching algorithm to eliminate the noise and extract the core content proposed by Uma and Latha [25] termed as *Exiting 2* are compared with the proposed method to prove the efficiency of the proposed method. Thus, these existing methods and the proposed method are executed for the set of web pages and the results produced are compared manually to compute the performance. The details such as the number of correctly extracted web ages and their percentage and the number of incorrectly extracted web pages and their percentage are given in Tab. 8.

Table 8. Performance Analysis of Core Content Extraction

No. of Web pages	Methods	No. of Correctly Extracted Web Pages	Correctly Extracted Web Pages (in %)	No. of Incorrectly Extracted Pages	Incorrectly Extracted Pages (in %)
100	Existing 1	90	90.00	10	10.00
	Existing 2	92	92.00	8	8.00
	Proposed	95	95.00	5	5.00
200	Existing 1	176	88.00	24	12.00
	Existing 2	181	90.50	19	9.50
	Proposed	184	92.00	16	8.00
300	Existing 1	269	89.67	31	10.33
	Existing 2	274	91.33	26	8.67
	Proposed	279	93.00	21	7.00

To analyse the overall effectiveness of the systems employed in the experimental analysis, the average values are computed for all the methods by dividing the sum of correctly extracted web pages by the sum of varied number of web pages used at each iteration. The average of correctly extracted web pages for the existing method 1, existing method 2 and the proposed method are 89.22%, 91.28% and 93.33%. Thus the proposed framework provides better efficiency in removing the primary noises such as advertisements, secondary noises such as duplicates and near duplicates and extracting core content block from the web pages. The accuracy of the methods is compared in Fig. 9.

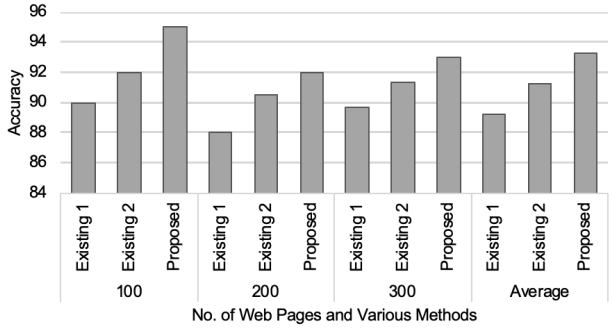

Fig. 9. Accuracy Comparison for Core Content Extraction

6. Conclusion

This paper focuses on removing the noises present in the web pages and extracting the significant core content from the web pages which is the way of eliminating the irrelevant and redundant contents from the web pages. This core content extraction and noise removal plays a vital role for many applications such as text categorization, page ranking, information retrieval and even help the users in accessing the interesting information. The paper categorizes the noises as primary noise and secondary noise. The primary noises such as advertisements, banners present in the web page are removed using the DOM tree and tag analysis. The secondary noises such as duplicates and near duplicates are identified and removed based on comparing the fingerprints generated by the proposed modified simhash algorithm that uses both frequency and proximity measure to calculate the weights for the terms. Finally, significant core contents are extracted by computing the weight based block score. Experimental and performance analysis has been made for the proposed method and the values are compared with the existing methods. The modified simhash algorithm gives better accuracy of 86.56% which is higher than several text similarity techniques such as Jaccard Coefficient, Minhash, Simhash and Hybridhash algorithms. The overall average accuracy of the proposed method is 93.33% which is also higher than other existing methods. The future work aims at mining the relevant images from the web based on the user query. The challenge is in recognizing procedures for building models that influence social media as well as news media will also be addressed.

References

1. J. Gadge and H. R. Parmar, Removal of Image Advertisement from Web Page. International Journal of Computer Applications, 27(7): 1–5 (2011).
2. Mayajohn, J. S. Jayasudha, Tag Based Noise Removal from Web Pages, Journal of Theoretical and Applied Information Technology, 95(22): 6336 – 6349 (2017).

3. R. Ghosh and S. Asur, Mining information from heterogeneous sources: A topic modeling approach. In Proc. of the MDS Workshop at the 19th ACM SIGKDD (2013).
4. B. Liu, and K. Chen-Chuan-Chang, Editorial: Special issue on web content mining. ACM SIGKDD Explorations Newsletter, 6(2): 1–4 (2004)
5. M. Agyemang, K. Barker and R. Alhajj, Framework for mining web content outliers. In Proc. of the ACM symposium on Applied computing, pp. 590-594 (2004).
6. G. Poonkuzhali, K. Thiagarajan, K. Sarukesi and G. V. Uma, Signed approach for mining web content outliers. World Academy of Science, Engineering and Technology, 56(9) (2009).
7. Y. Guo, H. Tang, L. Song, Y. Wang and G. Ding, ECON: an approach to extract content from web news page. In International Asia-Pacific Web Conference, IEEE, pp. 314-320 (2010).
8. S. Lingwal, Noise reduction and content retrieval from web pages. International Journal of Computer Applications, 73(4) (2013).
9. F. Johnson and S.K. Gupta, Web content mining techniques: a survey. International Journal of Computer Applications, 47(11) (2012).
10. S. Sirsat, Extraction of core contents from web pages. International Journal of Engineering Trends and Technology, 8(9): 484 - 489. (2014).
11. S. Zheng, R. Song and J. R. Wen, Template-independent news extraction based on visual consistency, In AAAI, 7: 1507-1513 (2007).
12. J. Prasad and A. Paepcke, Coreex: content extraction from online news articles, Proc. of the conference on Information and knowledge management. New York, NY, USA: ACM, pp. 1391–1392 (2008).
13. N. Pradhan, M. Gyanchandani and R. Wadhvani, A Review on Text Similarity Technique used in IR and its Application. International Journal of Computer Applications 120(9) (2015).
14. A. Huang, Similarity measures for text document clustering. In Proc. of the new zealand computer science research student conference, Christchurch, New Zealand, 4: 9-56 (2008).
15. S. Niwattanakul, J. Singthongchai, E. Naenudorn and S. Wanapu, Using of Jaccard coefficient for keywords similarity. In Proc. of the international multiconference of engineers and computer scientists, 1(6): 380-384 (2013).
16. C. H. Huang, J. Yin and F. Hou, A text similarity measurement combining word semantic information with TF-IDF method. Jisuanji Xuebao (Chinese Journal of Computers), 34(5): 856-864 (2011).
17. P. Lahoti, P. K. Nicholson, and B. Taneva, Efficient Set Intersection Counting Algorithm for Text Similarity Measures. In Proc. of the Workshop on Algorithm Engineering and Experiments. Society for Industrial and Applied Mathematics, p. 146-158 (2017).
18. A. Z. Broder, On the resemblance and containment of documents. In Proc. of the Compression and Complexity of Sequences, IEEE Computer Society, Washington, DC, USA, p. 21-29 (1997).
19. R. Pagh, M. Stöckel, and D. P. Woodruff, Is min-wise hashing optimal for summarizing set intersection?. In Proc. of the ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems, p. 109-120 (2014).
20. R. Hassanian-esfahani, and M. J. Kargar, Sectional minhash for near-duplicate detection. Expert Systems with Applications, 99: 203-212 (2018).
21. C. Sadowski and G. Levin, Simhash: Hash-based similarity detection, Technical report, Google (2007).
22. M. C. M. Xuguang, Research on Near-duplicate Detection Algorithm Shingling and Simhash, Computer & Digital Engineering, 1 (2009)
23. M. S. Uddin, C. K. Roy, K. A. Schneider, and A. Hindle, On the effectiveness of simhash for detecting near-miss clones in large scale software systems. In working conference on reverse engineering, IEEE p. 13-22 (2011).
24. P. Sivakumar, Effectual web content mining using noise removal from web pages. Wireless Personal Communications, 84(1) : 99-121 (2015).
25. R. Uma and B. Latha, Noise elimination from web pages for efficacious information retrieval, Cluster Computing, p.1-20 (2018).
26. S. S. Bama, M. I. Ahmed and A. Saravanan, Relevance Re-ranking Through Proximity Based Term Frequency Model, In International Conference on ICT Innovations, Springer, Cham, pp. 219-229 (2016).
27. S. S. Bama, M. I. Ahmed and A. Saravanan, A Mathematical Approach for Mining Web Content Outliers using Term Frequency Ranking. Indian Journal of Science and Technology, 8(14) (2015).
28. P. T. Ho, and S. R. Kim, Fingerprint-based near-duplicate document detection with applications to SNS spam detection. International Journal of Distributed Sensor Networks, 10(5), p.612970 (2014).
29. M. S. Charikar, Similarity estimation techniques from rounding algorithms. In Proc. of the thiry-fourth annual ACM symposium on Theory of computing, ACM, p. 380-388 (2002).
30. G. S. Manku, A. Jain and A. Das Sarma, Detecting near-duplicates for web crawling. In Proc. of the international conference on World Wide Web, ACM, pp. 141-150 (2007).