A Survey on Cleaning of Web Pages Before Web Mining

Neha Jagannath  
College of Engineering and Technology  
Bhubaneswar  
India

Sidharth Samant  
College of Engineering and Technology  
Bhubaneswar  
India

Subhalaxmi Das  
College of Engineering and Technology  
Bhubaneswar  
India

ABSTRACT
Web mining is one of the most essential tools in gathering the right information from the Internet and the World Wide Web. However, there are certain elements in a web page that hamper the web mining algorithms resulting in skewed output. These are called noises and include advertisements, navigation panels, copyright notices and other such things. It is thus essential that these noises are eliminated to make the web mining algorithms give a better output. This survey paper examines the different types of techniques that are available for cleaning local noises from webpages before web content or structure mining is performed, and compares their respective merits and demerits.

Keywords
Web Pages, Web Mining, WWW, Noise Cleaning, DOM, Entropy, Information Retrieval

INTRODUCTION
The Internet and the World Wide Web (WWW) have contributed immensely to the digital development of human civilization, by continually expanding upon the vast library of human knowledge. Web mining aims to discover useful knowledge from the Web through hyperlinks, page content and usage log [4]. But in this virtual world of bits and code, it has become colossally difficult to harness the right information from its anarchic mess. Proper techniques are needed in order to gather the required information from the Internet. However, this task is made even harder by the fact that unneeded information is often jumbled together with the needed.

The various additional features may help in enriching the user experience for the web site, besides performing some other functions, but they serve no purpose in the task of recovery of useful information from the Internet[14][15]. Instead, they hamper the algorithms[17] implemented for web mining tasks such as information retrieval and extraction, web page clustering and web page classification. These distractions are called “noise”, which hamper the retrieval of information and its subsequent use associated with web mining. Our survey is on the examination of said noise and on the techniques to eliminate them from the web pages in order to ensure more efficient performance by the web mining algorithms. Web cleaning is the process of detecting and eliminating noisy data from web pages before web mining in any form is done.

NOISES IN A WEB PAGE
Noise in a webpage can be defined as any section of the webpage which lends little to defining the contents of the web page. Since these noises contribute little to nothing in the results of web mining algorithms, they pose a serious hindrance to the accuracy of these algorithms and can cause unnecessary and unwanted results. Noise data of web documents can be grouped into two categories [3] according to their granularities:

- **Global noises**: These are noises on the Web with large granularity, which are usually no smaller than individual pages. Global noises include mirror sites, legal/illegal duplicated web pages, old versioned web pages to be deleted, etc.

- **Local (intra-page) noises**: These are noisy regions/items within a web page. Local noises are usually incoherent with the main contents of the web page. Such noises include:
  - Navigation: Intra and inter hyperlinks to guide the user to different part of a web page.
  - Decoration: Pictures, animations, logos and so forth for attraction purpose.
  - Interaction: Forms to collect user information or provide searching services, download links, etc.
  - Banner Advertisements: For promotion of either the home, or partner websites.
  - Other special words or paragraphs such as copyrights and contact information.

RELATED WORK
There have been various approaches by researchers for detection and removal of noise from web pages in order to increase the accuracy of the web mining algorithms which are similar to, or precursors of the techniques focused on. Some of these are: Evert [8] presents a new tool, NCLEANER - which aims to clean web pages with accurately using a pipeline of four modules in order to enable efficient extraction of web content as training data.
The first stage of the pipeline performs some normalization on the original HTML code. In the second stage, the preprocessed HTML page is converted to plain text using the text-mode browser Lynx.

In a third post-processing step, invalid characters are removed from the text, and some easily recognizable types of boilerplate deleted with regular expressions. The fourth and core component of the NCLEANER algorithm consists of two separate character-level n-gram language models for “clean” and “dirty” text in order to classify text segments as one or the other.

Qi and Sun [9] propose elimination of noisy information through heuristic rules by dividing the web into three categories, HUB pages, picture pages and topic pages and then parsing, filtering and eliminating noise using the DOM tree of each page.

Htwe and Kham [10] offer the approach of extracting the data region in a web page by removing noise using DOM and neural network.

LAMIS [11], standing for entropy-based Link Analysis on Mining web Informative Structures is an algorithm proposed by Kao et al. with the key idea being to utilize information entropy for representing the knowledge that corresponds to the amount of information in a link or a page in the link analysis, thereby mining web informative structures and discard noise.

Tripathy and Singh [12] propose a technique where a Pattern Tree is used to capture the general presentation styles and the definite essence of the pages in a specified web site. For a particular site, a pattern tree called the Site Pattern Tree (SPT) is generated by sampling the pages of the site. An information-based measure then decides which parts in the SPT represent noise, and which parts represent the core contents of the site. By mapping any web page to the SPT, the noises are detected and eliminated from that particular web page.

**TECHNIQUES ADOPTED**

This survey focuses on methods to eliminate local noises before web content or structure mining.

**Segmentation Based Cleaning Method:**

This method [1] employs the use of informative content blocks to detect the templates that are used to describe the contents of each section of the web page. It is a supervised learning method. The underlying principle is that a web page is made up of several templates; a template being a pre-written skeleton page that serves as a basis for creating new pages. So, every web site consists of these templates and the method automatically separates the informative content blocks from semantically redundant content such as advertisements, banners, and navigation panels. The distinguishing feature is that the redundant blocks in a page consist of the same design and presentation style as in other pages of the website. The discovery of informative content blocks happens in the following steps:

1. **Page Segmentation:** A content block is formed from every <TABLE> element in the DOM tree structure. The remaining contents also form a special block. Thus, this step essentially consists of extracting content blocks.

2. **Block Evaluation:** The feasible features of every content block are simultaneously selected and their corresponding entropy values are calculated. This step thus involves extraction and assessment of attributes of the content blocks.

3. **Block classification:** A specific optimal block threshold entropy value is estimated, which is to be used to differentiate the informative content blocks from the redundant blocks.

4. **Informative Block Detection:** Finally, the different blocks are classified as informative content blocks or redundant blocks based on whether their entropies are above or below the decided threshold.

A sample extraction process is shown in Figure 1.

![Figure 1. Extracting content blocks from a sample page.](image)

An improved version [2] not only considers HTML tables, but other tags too.

**Layout Analysis Based Cleaning Method:**

Following a different representation of page segmentation, this method [13] eliminates noisy information on a page layout basis. It presents an automatic top-down, tag-tree independent approach to detect web content structure which simulates how users understand web layout structure based on their visual perception. It consists of the use of the Vision-Based Page Segmentation Algorithm (VIPS) [15], followed by application of a set of heuristic rules to eliminate the noisy (non-coherent) blocks.

Generally, a web page designer would organize the content of a web page to make it easy for reading. Thus, as explained by Yang and Zhang [16], semantically related content is usually grouped together and the entire page is divided into regions for different contents using explicit or implicit visual separators such as lines, blank areas, images, font sizes, colors, etc. The goal of VIPS is to derive this content structure
from the visual representation of a web page in order to segment a web page while eliminating noise. The algorithm makes use of the layout features of the page and tries to partition the page at the semantic level. Each node in the extracted content structure corresponds to a block of coherent content in the original page.

After the segmentation process, a vision-based content structure of a web page is derived by combining the DOM tree and visual cues. Each node of the tree represents a region in the web page called a visual block. If a node has children in the structure, visual blocks of the children are contained within that of the node and form a partition of it. For each node, the Degree of Coherence (DoC) is defined to show how coherent it is. The value of DoC usually ranges from 0 to 1. The Permitted Degree of Coherence (PDoC) can be predefined to achieve different granularities of page segmentation for different applications. To obtain the vision-based content structure for a web page, the VIPS algorithm shown in Figure 2, is employed, consisting of the following steps:

1. **Visual Block Extraction:** Appropriate visual blocks in the current subtree are found.

   ```plaintext
   Algorithm DivideDOMtree(pRoot, nLevel)
   {
   IF (Dividable(pRoot, nLevel) == TRUE){
   FOR EACH child OF pRoot (
   DivideDOMtree(child, nLevel);
   }
   } ELSE {
   Put the sub-tree (pRoot) into the pool as a block;
   }
   }
   
   Algorithm Dividable(pRoot, nLevel)
   {
   IF (pRoot is the Top Block){
   RETURN TRUE;
   } ELSE {
   Special routines for TABLE, TR, TBODY, TD, P, UL, FORM;
   Heuristic rules for general tags;
   }
   }
   
   Figure 2. The Visual Block Extraction Algorithm
   ```

2. **Visual Separator Detection:** When all blocks are extracted, they are put into a pool for separator detection. Appropriate weight is set to each separator according to some patterns and those with highest weight are selected as the actual separators.

3. **Content Structure Construction:** When the actual separators are detected, visual blocks on the same sides of all the separators are merged and represented as a node in the content structure. After that, DoC of each node is calculated, and checks are done as to whether it meets the granularity requirement or not. For every node that fails to meet the requirement, the algorithm loops back to the visual block extraction step to further construct the sub content structure within the node. If and when all the nodes meet the requirement, the iterative process is stopped and the vision-based content structure for the whole page is obtained.

![Flowchart of the VIPS Algorithms](image)

**SST Based Cleaning Technique:**

This approach [3] is a "partially" supervised cleaning technique which employs the use of an SST (Site Style Tree) to detect and clean out noisy data. It analyses both the layouts and the actual contents of the web pages. The SST is a derivative of the DOM tree structure, but instead of consisting of the contents and layouts of the web page, it contains the style elements and the presentation styles of the elements in the web page. This is illustrated in Figure 4.

The algorithm, shown in Figure 5, works through the following steps:

1. A SST is constructed from the DOM tree. This contains the common layouts and presentation styles present in the web page.
2. Information entropy values are evaluated for each node of the SST, and an optimal threshold entropy value is decided upon to differentiate noisy data from meaningful data.
3. Using the threshold entropy, noisy data is detected and eliminated.

![DOM trees and the Style Tree](image)
1. Randomly crawl $k$ pages from the given Web site $S$.
2. Set null SST with virtual root $E$ (representing the root).
3. For each page $W$ in the $k$ pages do
   4. BuildPST($W$).
   5. BuildSST($E, E_0$).
4. end for.
5. CalcCompImp($E$).
7. MarkMeaningful($E$).
8. For each target Web page $P$ do
9.   $E_p$: BuildPST($P$) /* representing the root */
10.  MapSST($E, E_p$).
11. end for.

Figure 5. The Overall SST-based Cleaning Algorithm

Feature Weighting Based Cleaning Method:
This is an improved version of the SST based cleaning method, developed by Yi and Liu [4]. It is unsupervised and employs the use of a CST (Compressed Structure Tree) instead of an SST, as demonstrated in Figure 6. The following revisions are made by this algorithm:
1. Elements within the tree of every document are combined (compressed) if their child elements share the identical tag names, attributes, and attribute values.
2. Based on the number of different presentation styles for an element, the weight (importance) of this element is determined using entropy calculation.
3. The resulting weights are then utilized in follow-up tasks (e.g. classification).

Template Based Cleaning Method:
Many websites, especially those which are professionally designed, consist of templated pages. A templated page (Figure 8) is one among a number of pages sharing a common administrative authority and a look and feel. The shared look and feel is very valuable from a user’s point of view since it provides context for browsing. However, templated pages skew ranking. Information Retrieval (IR) and Data Mining (DM) algorithms and consequently, reduce precision.

This technique [5] is an unsupervised and automatic method which relies on templates to clean noisy data from web pages. For the detection of templates, the algorithm first partitions the web page into pagelets. A pagelet (Figure 7) can be defined as a self-contained logical region within a web page which has a well-defined topic or functionality (and is not nested within another region that has exactly the same topic or functionality, being self-contained).

The algorithm places emphasis on pagelets rather than pages because pagelets are more structurally cohesive, adhere better to a single, common topic and generally point to relevant links. It works through the following steps:

1. The web page is partitioned into logically articulate pagelets that maintain topical unity through a page partitioning algorithm (Figure 9). This partition is fixed.
according to the number of hyperlinks that the HTML element has.

\[
\text{Page Partitioning Algorithm}
\]
\[
\text{Partition}(p) \{
T_p \leftarrow \text{HTML parse tree of } p
\text{Queue} \leftarrow \text{root of } T_p
\text{while (Queue is not empty)} \{
\text{v} \leftarrow \text{top element in Queue}
\text{if (v has a child with at least k links)}
\text{push all the children of } v \text{ to Queue}
\text{else}
\text{declare } v \text{ as a pagelet}
\}
\]

Figure 9. The Page Partitioning Algorithm

2. Frequently occurring templates within the pagelets are detected. These are the required noisy data. Two kinds of algorithms are used for template detection. The local template detection algorithm (Figure 10) is suitable for the document sets that consist of small fraction of documents from the larger universe, while the global template detection algorithm (Figure 11) is suitable for template detection in large subsets of the universe. It requires the detected templates to be undirected connected by hyperlinks.

\[
\text{Local Template Detection Algorithm}
\]
\[
\text{(1) Eliminate duplicate pages in } G
\text{(2) Sort and group the pagelets in } G \text{ according to their shingle. Each such group represents a template.}
\text{(3) Enumerate the groups, and output the pagelets belonging to each group.}
\]

Figure 10. The Local Template Detection Algorithm

\[
\text{Global Template Detection Algorithm}
\]
\[
(1) \text{ Select all the pagelet shingles in \text{PAGELIT} that have at least two occurrences. Call the resulting table \text{TOPPLAYJUNKIE}. These are the shingles of the re-occurring pagelets.}
(2) \text{ Extract from \text{PAGELIT} only those pagelets whose shingle occurs in \text{TOPPLAYJUNKIE}. Call the resulting table \text{TEMPLATEJUNKIES}. These are all the pagelets that have multiple occurrences in } G.
(3) \text{ For every shingle } s \text{ that occurs in } \text{TEMPLATEJUNKIES} \text{ define } G_s \text{ to be the shingle's groups: all the pages that contain pagelets whose shingle is } s. \text{ By joining } \text{TEMPLATEJUNKIES} \text{ and } \text{LINKS} \text{ find for every } s \text{ all the links between pages in } G_s. \text{ Call the resulting relation } \text{TEMPLATEJUNKIES}.
(4) \text{ Enumerate the shingles in } \text{TEMPLATEJUNKIES}. \text{ For each one load into main memory all the links between pages in } G_s.
(5) \text{ Use a BFS algorithm to find all the undirected connected components in } G_s. \text{ Each such component is either a template or a singleton. Output the component if it is not a singleton.}
\]

Figure 11. The Global Template Detection Algorithm

Classification Based Cleaning Method:
This method uses pattern classification techniques, employing pre-processing to clean and reduce the link dataset before calculations are performed by detecting and eliminating specific noisy data, such as advertisements or nepotistic/navigational links. It is supervised and semi-automatic. There can be a variety of classification based cleaning method based on what noisy data the user wants to remove. But the common underlying principle remains the same, which is to use a decision tree classifier to detect noisy data.

The Decision Tree Classifier poses a series of carefully crafted questions about the attributes of the test record. Each time it receives an answer, a follow-up question is asked until a conclusion about the class label of the record is reached. Some of the decision trees used are ID3 (Iterative Dichotomiser 3), CART and C4.5. Of these, C4.5 is most widely utilized because of its value as a popular and well-understood learning system.

A decision tree classification algorithm can be adopted according to the user requirements in order to concentrate upon a specific type of noisy data, such as advertisements or navigational bars. An example decision tree has been shown in Figure 12.

Figure 12. Example Decision Tree based on provided data

The algorithm work in the following steps:
1. The features and characteristics that define the targeted noisy data are specified.
2. A decision tree is built based on sample items (both noisy and non-noisy) and rules are extracted through the inductive learning algorithm that processes the training examples to generate a set of rules.
3. Based upon these extracted rules, the algorithm differentiates between noisy and non-noisy data. For example, AdEater [6] is a browsing assistant based on this system that automatically removes banner advertisements from Internet pages before the corresponding images are downloaded. Paek’s work [7] trains the decision tree classifier to recognize banner advertisements, while Davison’s work [17] recognizes nepotistic links.

**COMPARISON**

The noise patterns across different websites are varied and non-uniform, and hence, difficult to detect [14]. Each of the cleaning techniques discussed above have some characteristics that specify the kind of web page they are suitable for, the kind of noise they can target, and so on. The following is an assessment of the various features that define these methods and set them apart from each other:

<table>
<thead>
<tr>
<th>Cleaning Methods</th>
<th>Supervision</th>
<th>Automation</th>
<th>Basis</th>
<th>Suitability</th>
<th>Review</th>
</tr>
</thead>
<tbody>
<tr>
<td>Segmentation-Based</td>
<td>Supervised</td>
<td>Not automatic (the system must know a priori how a web page can be partitioned into coherent content blocks, and which blocks are the same blocks in different web pages)</td>
<td>Informative blocks</td>
<td>News web sites (the improved method can be applied to other websites too, but is still most suitable for news web sites)</td>
<td>Based on the assumption of knowing page clusters.</td>
</tr>
<tr>
<td>Layout Analysis-Based</td>
<td>Supervised</td>
<td>Automatic</td>
<td>Vision-based Content Structure</td>
<td>Site independent and independent to the HTML documentation representation of a web page</td>
<td>Works well even when the HTML structure is quite different from the visual layout structure. But, the realization of VIPS algorithm is complex and time expensive, and some of the current implementations of the algorithm are based on IE kernel, so there are some practical limitations.</td>
</tr>
<tr>
<td>SST-Based</td>
<td>Partially Supervised</td>
<td>Automatic (assumptions made for segmentation are automatic)</td>
<td>DOM tree/ Site Style Tree</td>
<td>Web pages having similar presentation styles</td>
<td>Construction of SST requires learning the whole web site to find out common presentation styles. Not useful when noises are of different presentation styles such as in dynamic web pages, and less successful in detecting noise patterns that vary from expected patterns.</td>
</tr>
<tr>
<td>Feature Weighing-Based</td>
<td>Unsupervised</td>
<td>Automatic</td>
<td>DOM tree/ Compressed Style Tree</td>
<td>Website, even if distinct, must have common presentation styles</td>
<td>Depends on the availability of a large number of documents from a limited number of sources.</td>
</tr>
</tbody>
</table>
CONCLUSION

As the web continues to expand, so does the amount of noise, and it is that much more crucial to discover a technique to counteract this growing rise of unneeded data in web pages. The task of web mining can be made much easier and the results much more accurate, with the usage of the appropriate technique.

In this paper, we have examined different techniques to eliminate noise in web mining. However, none of these techniques can be implemented to eliminate every noise present in a website. While some suffer from the problem of site dependency, others work only on specific noises, and still others have complex algorithms that are hard to implement. There is still ongoing research on web mining and noise elimination and there is hope for the realization of a method that can eliminate any type of noise that is encountered.

REFERENCES


