HTML Web Content Extraction Using Paragraph Tags

Howard J. Carey, III, Milos Manic
Department of Computer Science
Virginia Commonwealth University
Richmond, VA USA
careyhj@vcu.edu, misko@ieee.org

Abstract— With the ever expanding use of the internet to disseminate information across the world, gathering useful information from the multitude of web page styles continues to be a difficult problem. The use of computers as a tool to scrape the desired content from a web page has been around for several decades. Many methods exist to extract desired content from web pages, such as Document Object Model (DOM) trees, text density, tag ratios, visual strategies, and fuzzy algorithms. Due to the multitude of different website styles and designs, however, finding a single method to work in every case is a very difficult problem. This paper presents a novel method, Paragraph Extractor (ParEx), of clustering HTML paragraph tags and local parent headers to identify the main content within a news article. On websites that use paragraph tags to store their main news article, ParEx shows better performance than the Boilerpipe algorithm with higher F1 scores of 97.33% to 88.53%.

Keywords—HTML, content extraction, Document Object Model, tag-ratios, tag density.

I. INTRODUCTION

The Internet is an ever growing source of information for the modern age. With billions of users and countless billions of web pages, the amount of data available to a single human being is simply staggering. Attempting to process all of this information is a monumental task. Vast amounts of information that may be important to various entities are provided in web based news articles. Modern day news websites are updated multiple times a day making more data constantly available. These web based articles offer a good source for information because of the relatively free availability, ease of access, large amount of information, and ease of automation. To analyze news articles from a paper source, the articles would first have to be read into a computer, making the process of extracting the information within much more cumbersome and time consuming. Due to the massive amount of news articles available, there is simply too much data for a human to manually determine which of this information is relevant. Thus, automating the extraction of the primary article is necessary to allow further data analysis on any information within a web page.

To help analyze the content of web pages, researchers have been developing methods to extract the desired information from a web page. A modern web page now a days consists of numerous links, advertisements, and various navigation information. This extra information may not be relevant to the main content of the web page, and can be ignored in many cases. This additional information, such as ads, can also lead to misleading or incorrect information being extracted. Thus, determining the relevant main content of a web page among the extra information is a difficult problem.

Numerous attempts have been made in the past two decades to filter the main content of a web page. Therefore, this paper presents Paragraph Extractor (ParEx), a novel method used to identify the main text content within an article on a website while filtering out as much irrelevant information as possible. ParEx relies upon using HTML paragraph tags, denoted by \( \langle p \rangle \) in HTML, combined with clustering and entity relationships to extract the main content of an article. It was shown that ParEx had very high recall and precision scores on a set of test sites that use \( \langle p \rangle \) tags to store their main article content and have little to no user comment section.

This rest of this paper is organized in the following format: Section II examines related works in the field. Section III details the design methodology of ParEx. Section IV discusses the evaluation metrics used to test the method. Section V describes the experimental results of ParEx. And finally, section VI summarizes the findings of this paper.

II. RELATED WORKS

Early attempts at content extraction mostly had some sort of human interaction required to identify the important features of a website, such as [1],[9],[10]. While these methods could be accurate, they were not easily scalable to bulk data collection. Other early methods employed various natural language processing methods [7] to help identify relationships between zones of a web page, or utilized HTML tags to identify various regions within the text [14]. Kushmerick developed a method to solely identify the advertisements in a page, and remove them [11].

Many methods attempt to utilize the Document Object Model (DOM) to extract formatted HTML data from websites [3, 4]. DOM provides “a platform and language-neutral interface that allows programs and scripts to dynamically access and update the content, structure and style of
documents.” [13]. In [2], Gongquing et al. utilized a DOM tree to help determine a text-to-tag path ratio, while in [20] Mantratzis et al. developed an algorithm that recursively searches through a DOM tree to find which HTML tags contained a high density of hyperlinks.

Layouts can be common throughout web pages in the forms of templates. Detection of these templates and the removal of the similar content that occurs between multiple web pages can leave the differing content between them, which can be the main article itself, as found by Bar-Yossef et al. in [22]. Chen et al. in [15] explored a method of combining layout grouping and word indexing to detect the template. In [16] and [23], Kao et al. developed an algorithm that utilized the entropy of feature, links, and content of a website to help identify template sections. And Yi et al. in [17] proposed a method to classify and cluster web content using a style tree to help compare between website structures to determine the template used. Kohlschütter [30] developed a method, Boilerpipe, to detect shallow text features in templates to help detect the boilerplate (any section of website which is not considered main content) using the number of words and link density of a website.

Much research tends to build on the work from previous researchers. In [5], Gottron provided a comparison between many of the content extraction algorithms at the time, and modified the document slope curve algorithm from [18]. The modified document slope curve proved the best within his test group. In [18], Pinto et al. expanded on the work of Body Text Extraction from [14] by utilizing a document slope curve to identify content vs. non-content pages in hopes of determining whether a web page had content worth extracting or not. Debnath et al. proposed the algorithms ContentExtractor [21] and FeatureExtractor [22], that compared similarity between blocks across multiple web pages and classified sections as content with respect to a user defined desired feature set.

In [19], Spousta et al. developed a cleaning algorithm that involved regular expression analysis and numerous heuristics involving sentence structure. Their results performed poorly on web pages with poor formatting or numerous images and links. Gottron [24] utilized a content code blurring technique to identify the main content of an article. The method involved applying a content code ratio to different areas of the web page and analyzing the amount of text in the different regions.

Many recent works have taken newer approaches, but still tend to build on the works of previous research. In [12], Bu et al. proposed a method to analyze the main article using fuzzy association rules (FAR). They encoded the min., mean, and max values of all items and features for a web page into a fuzzy set and achieved decent, but quick, results. Song et al., in [6] and Sun et al. [8], expanded on the tag path ratios by looking at text density within a line and taking into account the number of all hyperlink characters in a subtree compared to all hyperlink tags in a subtree. Peters et al. [25] combine elements of Kohlschutter’s boilerplate detection methods, [30], and Weninger’s CETR methods, [3] and [4], into a single machine learning algorithm. Their combined methodology showed improvements over using just a single algorithm, however had trouble with websites that used little CSS for formatting.

Nethra et al. [26] created a hybrid approach using feature extraction and decision tree classification. They used C4.5 decision tree and Naïve Bayes classifiers to determine which features were important in determining main content. In [27], Bhardwaj et al. proposed an approach of combining the word to leaf ratio (WLR) and the link attributes of nodes. Their WLR was defined as the ratio between the number of words in a node to the number of leaves in the subtree of said node. In [28], Gondse et al. proposed a method of extracting content from unstructured text. Using a web crawler combined with user input to decide what to look for, the crawler analyzes the DOM tree of various sites to find potential main content sections. Qureshi et al. [29] created a hybrid model utilizing a DOM tree to determine the average text size and link density of each node.

No algorithm has managed to achieve 100% accuracy in extracting all relevant text from websites so far. With the ever changing style and design of modern web pages, different approaches are continually needed to keep up with the changes. Some algorithms may work on certain websites but not others. There is much work left to be done in the of website content extraction.

III. PAREx WEB Page CONTENT EXTRACTION METHOD

The steps involved in ParEx are shown in Figure 1. The Preprocessing step starts with the original website’s HTML being extracted and parsed via JSoup [32]. The Text Extraction section locates and extracts the main content within the website. The Final Text is then analyzed using techniques elaborated on in section IV.

A. Preprocessing

The HTML code was downloaded directly from each website and parsed using the JSoup API [32] to allow easy filtering of HTML data. In this way, HTML tags were pulled out, processed and extracted from the original HTML code. JSoup also simplified the process of locating tags and parent

![Figure 1: Flow diagram of the presented ParEx method.](image-url)
be assigned a clustered ratio that is the average of lines 2-6, 32.8. This process is repeated for all lines in the raw HTML, before any other formatting is done. This clustering helps filter out one line advertisements or extraneous information that have high text-to-tag ratios while favoring clusters of high text-to-tag ratios.

C. Parent Header

The highest relative tag ratio will help to find the single sentence that is most likely to contain the main article text within it. This extracted line’s parent header is taken to be the master header for the web page. All \texttt{\langle p \rangle} tags under this master header are then extracted and considered to be the primary article text within the web page. Examples of this procedure are shown in Figures 3 and 4 and discussed below.

In Figure 3, there are three \texttt{\langle p \rangle} tags. If line 3 has the greatest relative text-to-tag ratio, it is extracted as the most likely to contain the main content. The parent of line 3 is found as \texttt{\langle local \rangle}. Then, all \texttt{\langle p \rangle} tags under \texttt{\langle local \rangle} are extracted and considered to be the main content. While \texttt{\langle main \rangle} is the most senior parent in the tree, only the immediate parent is chosen when determining the parent of the relative \texttt{\langle p \rangle} tag. Selecting a more senior parent generally selects more information than the desired content. Also in this case, no tags under \texttt{\langle next \rangle} would be chosen.

Lists in HTML are often used to list out various talking points within an article, however these are not generally formatted into \texttt{\langle p \rangle} tags, but into list tags, denoted by \texttt{\langle li \rangle}. If any \texttt{\langle li \rangle} tags are found within the master header, then their text contents are added to the extracted article text as well.

In Figure 4, if the \texttt{\langle p \rangle} tag on line 3 was chosen as the main content containing line, then all \texttt{\langle p \rangle} tags under the \texttt{\langle local \rangle}

\begin{verbatim}
1 <main>
2   <local>
3     <p> This is paragraph one. </p>
4     <p> This is paragraph two. </p>
5   </local>
6   <next>
7     <p> This is paragraph three. </p>
8   </next>
9 </main>
\end{verbatim}

Figure 3: \texttt{\langle p \rangle} tag extraction example.
tag would be extracted. However, information within the list tags, \( \langle li \rangle \), would also be extracted, as the algorithm recognizes any \( \langle p \rangle \) as well as \( \langle li \rangle \) tags under the chosen master parent tag.

Once the final list of \( \langle p \rangle \) and \( \langle li \rangle \) tags are extracted, the text of these tags is compared to the original, annotated data that was extracted using methods discussed in section A. The techniques used to evaluate the effectiveness of the algorithm are discussed in the next section of the paper.

IV. EVALUATION METRICS

Three metrics are used to evaluate the performance of the ParEx, precision, recall, and F1 score. These are standard evaluation techniques used to determine how accurate an extracted block of text is compared to what it is supposed to be.

Precision (\( P \)) is a ratio between the size of the intersection of the extracted text and the actual text, over the size of the extracted text. Precision gives a measurement of how relevant the extracted words are to the actual desired content text and is expressed as:

\[
P = \frac{S_E \cap S_A}{S_E}
\]  

(2)

where \( S_E \) is the extracted text from the algorithm and \( S_A \) is the actual article text.

Recall (\( R \)) is a ratio between the size of the intersection of the extracted text and the actual text, over the size of the actual text. Recall gives a measurement of how much of the relevant data was accurately extracted and is expressed as:

\[
R = \frac{S_E \cap S_A}{S_A}
\]

(3)

where \( S_E \) is the extracted text from the algorithm and \( S_A \) is the actual article text that serves as the base line comparison for accuracy.

The F1 (\( F \)) score is a combination of precision and recall, allowing for a single number to represent the accuracy of the algorithm. It is expressed as:

\[
F = \frac{2 \times P \times R}{P + R}
\]

(3)

where \( P \) is the precision and \( R \) is the recall.

These metrics analyze the words extracted from each algorithm and compare them to the words in the article itself. If the words extracted from the algorithm closely match the actual words in the article, the F1 score will be higher. Using these three scoring methods allows for a numerical comparison between various algorithms. F1 score is important because looking purely at either precision or recall can be misleading. Having a high precision but low recall value means nearly all of the extracted data was accurate, however it does not tell any information on how much relevant information is missing. Having a high recall but low precision value means nearly all of the data that should have been extracted was, but it does not tell any information on how much extra, non-relevant data was also extracted. Thus, the F1 score is useful as it allows a combination of both precision and recall to determine a balance between the two.

ParEx was tested against manually extracted data from each website. This allowed the results to be verified against a known dataset. The precision, recall, and F1 score were obtained by using the set of words that appeared in the manually extracted text and comparing against the set of words extracted by the tested algorithms. No importance was given to the order in which certain words appeared.

V. EXPERIMENTAL RESULTS

The hypothesis being tested was whether or not the ParEx method would be a more effective extraction algorithm than Boilerpipe on websites which used \( \langle p \rangle \) tags to store their main content and had little to no user comment sections. To test this hypothesis, 15 websites were selected which exhibited these characteristics and 15 websites which did not exhibit these characteristics were selected.

The websites selected were mostly local news based websites, such as FOX, CBS, NBC, NPR, etc., with a subject focus on critical infrastructure events in local areas. All 30 of the articles selected were from different websites. The wide variety of websites allowed both methods to be tested on as diverse a set of sites as possible. The use of news article sites as opposed to other sites, such as blogs, forums, or shopping sites, is because news sites tend to have a single primary article of discussion per site. To acquire HTML data form these sites, it was simply downloaded from the website directly. To compare the algorithm’s extracted text with the actual website main content text, the main content needed to be extracted and annotated. The annotation was done manually, by a human. The final decision about what was considered relevant to the main article within the website was up to the annotator.
This method of manual content extraction for comparison has some inherent error risk involved. Different people may consider different parts of an article as main content. For instance, the title and header of the article may be viewed as main content or not. The author information at the bottom of an article may or may not be viewed as main content by different people. For the purposes of this paper, only the main news article content was considered, leaving off any author information, contact information, titles, or author notes. With this in mind, a certain amount of error is to be expected even with accurate algorithm results.

The accuracy of ParEx was compared against the accuracy of the Boilerpipe tool, developed by Kohlschutter [30]. Boilerpipe is a fully developed tool that is publically available and provides numerous methods of content extraction via an easy to use API [31] allowing easy testing and performs well on a broad range of website types.

Table I shows ParEx performed better than Boilerpipe in all metrics for websites that exhibited the required characteristics. Table II shows Boilerpipe performed much better than ParEx on websites that did not exhibit the required characteristics. These results support the original assumption, for websites that use paragraph tags to store their main content and have limited to no user comment sections ParEx will have a higher performance.

Examining the differences of each method between both tables I and II shows that while ParEx may be more accurate on sites that exhibit the required characteristics, Boilerpipe is a more generalizable method. ParEx claims an F1 score of 97.33% on the first data set, while only a 33.73% on the second. Boilerpipe, however, achieves F1 scores of 88.53% on the first set, and 82.53% on the second. Boilerpipe does not work as well on sets that exhibit the characteristics required for ParEx, but it performs more consistently on multiple types of websites.

<table>
<thead>
<tr>
<th>Method</th>
<th>ParEx</th>
<th>Boilerpipe</th>
</tr>
</thead>
<tbody>
<tr>
<td>Metric</td>
<td>Scores</td>
<td>Standard Deviation</td>
</tr>
<tr>
<td>Precision</td>
<td>96.07%</td>
<td>3.53%</td>
</tr>
<tr>
<td>Recall</td>
<td>98.73%</td>
<td>1.19%</td>
</tr>
<tr>
<td>F1 Score</td>
<td>97.33%</td>
<td>2.04%</td>
</tr>
</tbody>
</table>

Table II: Score comparison between both methods on the set of non-<p> tagged sites.

<table>
<thead>
<tr>
<th>Method</th>
<th>ParEx</th>
<th>Boilerpipe</th>
</tr>
</thead>
<tbody>
<tr>
<td>Metric</td>
<td>Scores</td>
<td>Standard Deviation</td>
</tr>
<tr>
<td>Precision</td>
<td>40.33%</td>
<td>38.36%</td>
</tr>
<tr>
<td>Recall</td>
<td>41.53%</td>
<td>40.84%</td>
</tr>
<tr>
<td>F1 Score</td>
<td>33.73%</td>
<td>34.52%</td>
</tr>
</tbody>
</table>

Figure 5: Individual website results for the <p> tagged website set.

Figure 6: Individual website results for the non-<p> tagged website set.

Examining the differences of each method between both tables I and II shows that while ParEx may be more accurate on sites that exhibit the required characteristics, Boilerpipe is a more generalizable method. ParEx claims an F1 score of 97.33% on the first data set, while only a 33.73% on the second. Boilerpipe, however, achieves F1 scores of 88.53% on the first set, and 82.53% on the second. Boilerpipe does not work as well on sets that exhibit the characteristics required for ParEx, but it performs more consistently on multiple types of websites.

Examining the differences of each method between both tables I and II shows that while ParEx may be more accurate on sites that exhibit the required characteristics, Boilerpipe is a more generalizable method. ParEx claims an F1 score of 97.33% on the first data set, while only a 33.73% on the second. Boilerpipe, however, achieves F1 scores of 88.53% on the first set, and 82.53% on the second. Boilerpipe does not work as well on sets that exhibit the characteristics required for ParEx, but it performs more consistently on multiple types of websites.

This method of manual content extraction for comparison has some inherent error risk involved. Different people may consider different parts of an article as main content. For instance, the title and header of the article may be viewed as main content or not. The author information at the bottom of an article may or may not be viewed as main content by different people. For the purposes of this paper, only the main news article content was considered, leaving off any author information, contact information, titles, or author notes. With this in mind, a certain amount of error is to be expected even with accurate algorithm results.

The accuracy of ParEx was compared against the accuracy of the Boilerpipe tool, developed by Kohlschutter [30]. Boilerpipe is a fully developed tool that is publically available and provides numerous methods of content extraction via an easy to use API [31] allowing easy testing and performs well on a broad range of website types.

Table I shows ParEx performed better than Boilerpipe in all metrics for websites that exhibited the required characteristics. Table II shows Boilerpipe performed much better than ParEx on websites that did not exhibit the required characteristics. These results support the original assumption, for websites that use paragraph tags to store their main content and have limited to no user comment sections ParEx will have a higher performance.

Examining the differences of each method between both tables I and II shows that while ParEx may be more accurate on sites that exhibit the required characteristics, Boilerpipe is a more generalizable method. ParEx claims an F1 score of 97.33% on the first data set, while only a 33.73% on the second. Boilerpipe, however, achieves F1 scores of 88.53% on the first set, and 82.53% on the second. Boilerpipe does not work as well on sets that exhibit the characteristics required for ParEx, but it performs more consistently on multiple types of websites.

Examining the differences of each method between both tables I and II shows that while ParEx may be more accurate on sites that exhibit the required characteristics, Boilerpipe is a more generalizable method. ParEx claims an F1 score of 97.33% on the first data set, while only a 33.73% on the second. Boilerpipe, however, achieves F1 scores of 88.53% on the first set, and 82.53% on the second. Boilerpipe does not work as well on sets that exhibit the characteristics required for ParEx, but it performs more consistently on multiple types of websites.

This method of manual content extraction for comparison has some inherent error risk involved. Different people may consider different parts of an article as main content. For instance, the title and header of the article may be viewed as main content or not. The author information at the bottom of an article may or may not be viewed as main content by different people. For the purposes of this paper, only the main news article content was considered, leaving off any author information, contact information, titles, or author notes. With this in mind, a certain amount of error is to be expected even with accurate algorithm results.

The accuracy of ParEx was compared against the accuracy of the Boilerpipe tool, developed by Kohlschutter [30]. Boilerpipe is a fully developed tool that is publically available and provides numerous methods of content extraction via an easy to use API [31] allowing easy testing and performs well on a broad range of website types.

Table I shows ParEx performed better than Boilerpipe in all metrics for websites that exhibited the required characteristics. Table II shows Boilerpipe performed much better than ParEx on websites that did not exhibit the required characteristics. These results support the original assumption, for websites that use paragraph tags to store their main content and have limited to no user comment sections ParEx will have a higher performance.

Examining the differences of each method between both tables I and II shows that while ParEx may be more accurate on sites that exhibit the required characteristics, Boilerpipe is a more generalizable method. ParEx claims an F1 score of 97.33% on the first data set, while only a 33.73% on the second. Boilerpipe, however, achieves F1 scores of 88.53% on the first set, and 82.53% on the second. Boilerpipe does not work as well on sets that exhibit the characteristics required for ParEx, but it performs more consistently on multiple types of websites.

Examining the differences of each method between both tables I and II shows that while ParEx may be more accurate on sites that exhibit the required characteristics, Boilerpipe is a more generalizable method. ParEx claims an F1 score of 97.33% on the first data set, while only a 33.73% on the second. Boilerpipe, however, achieves F1 scores of 88.53% on the first set, and 82.53% on the second. Boilerpipe does not work as well on sets that exhibit the characteristics required for ParEx, but it performs more consistently on multiple types of websites.

This method of manual content extraction for comparison has some inherent error risk involved. Different people may consider different parts of an article as main content. For instance, the title and header of the article may be viewed as main content or not. The author information at the bottom of an article may or may not be viewed as main content by different people. For the purposes of this paper, only the main news article content was considered, leaving off any author information, contact information, titles, or author notes. With this in mind, a certain amount of error is to be expected even with accurate algorithm results.

The accuracy of ParEx was compared against the accuracy of the Boilerpipe tool, developed by Kohlschutter [30]. Boilerpipe is a fully developed tool that is publically available and provides numerous methods of content extraction via an easy to use API [31] allowing easy testing and performs well on a broad range of website types.

Table I shows ParEx performed better than Boilerpipe in all metrics for websites that exhibited the required characteristics. Table II shows Boilerpipe performed much better than ParEx on websites that did not exhibit the required characteristics. These results support the original assumption, for websites that use paragraph tags to store their main content and have limited to no user comment sections ParEx will have a higher performance.

Examining the differences of each method between both tables I and II shows that while ParEx may be more accurate on sites that exhibit the required characteristics, Boilerpipe is a more generalizable method. ParEx claims an F1 score of 97.33% on the first data set, while only a 33.73% on the second. Boilerpipe, however, achieves F1 scores of 88.53% on the first set, and 82.53% on the second. Boilerpipe does not work as well on sets that exhibit the characteristics required for ParEx, but it performs more consistently on multiple types of websites.

The accuracy of ParEx was compared against the accuracy of the Boilerpipe tool, developed by Kohlschutter [30]. Boilerpipe is a fully developed tool that is publically available and provides numerous methods of content extraction via an easy to use API [31] allowing easy testing and performs well on a broad range of website types.

Table I shows ParEx performed better than Boilerpipe in all metrics for websites that exhibited the required characteristics. Table II shows Boilerpipe performed much better than ParEx on websites that did not exhibit the required characteristics. These results support the original assumption, for websites that use paragraph tags to store their main content and have limited to no user comment sections ParEx will have a higher performance.

Examining the differences of each method between both tables I and II shows that while ParEx may be more accurate on sites that exhibit the required characteristics, Boilerpipe is a more generalizable method. ParEx claims an F1 score of 97.33% on the first data set, while only a 33.73% on the second. Boilerpipe, however, achieves F1 scores of 88.53% on the first set, and 82.53% on the second. Boilerpipe does not work as well on sets that exhibit the characteristics required for ParEx, but it performs more consistently on multiple types of websites.
content, 2) the method is susceptible to comment sections, or other large blocks of text that may fool the clustering algorithm by selecting the wrong block of text as the main content.

VI. CONCLUSION

This paper presented a new method called ParEx to evaluate the content within a website and extract the main content text within. Building upon previous work with the text-to-tag ratio and clustering methods, the presented ParEx method focused only on the paragraph tags of each website, making the assumption that most websites will have their main article within a number of \( p \) tags. Two primary requirements were found to optimize the success of ParEx: 1) websites must use \( p \) tags to store their article content, 2) use websites that have limited or no user comment sections. The results showed that the ParEx method showed overall better performance than Boilerpipe on websites that exhibited these characteristics (with F1 scores of 97.33% vs. 88.53% for the Boilerpipe). This confirms the requirement for the ParEx approach.

Future work includes further improving the content extraction accuracy by improving the clustering algorithm and text-to-tag ratio metric to increase the likelihood that the algorithm will select the correct chunk of \( p \) tags to extract as the main content as opposed to a user comment section. Comparing the differences between a tag ratio that uses characters to one that uses words can also be explored.

ACKNOWLEDGMENT

The authors would like to thank support given by Mr. Ryan Hruska of the Idaho National Laboratory (INL) who helped make this effort a success.

REFERENCES


