Text Extraction from the Web via Text-to-Tag Ratio

Tim Weninger and William H. Hsu
Department of Computing and Information Sciences
Kansas State University
213 Nichols Hall
Manhattan, KS 66506 USA

Abstract – **We describe a method to extract content text from diverse Web pages by using the HTML document's Text-to-Tag Ratio rather than specific HTML cues that may not be constant across various Web pages. We describe how to compute the Text-to-Tag Ratio on a line-by-line basis and then cluster the results into content and non-content areas. With this approach we then show surprisingly high levels of recall for all levels of precision, and a large space savings.**

I. Introduction

The amount of information being gathered and stored on the Internet continues to increase. The artifacts of this growing market provide interesting new research opportunities that explore social interactions, language, art, mathematics, etc. Many of these new research opportunities require the content of the Internet to be gathered, processed and stored quickly and efficiently. This effort is often hampered by the use of structure tags in HTML and XML. These tags are meaningful only to the browser that renders the document, but bear little semantic meaning to the end user. Tags and other non-content related HTML characters – images not included – comprise the majority of each page's size [1], and yet, Internet researchers are forced to crawl, compute and store web content in their entirety.

Therefore, entire Web pages are needlessly downloaded and indexed. In order to save space and increase the accuracy of indexing, NLP techniques, etc. researchers have devised ways to extract only the content of a Web page while removing navigation links, advertisements and other miscellaneous text [2, 3]. These recent text extraction techniques attempt to glean content by looking for structural cues in the HTML document as described in Section II. We contend that these techniques are not only limited in their ability to extract information, but that their performance will be further deprecated by the separation of form from content brought on by the increasing popularity of cascading style sheets (CSS) [4, 5].

This work focuses on extracting content from Web pages that are otherwise laden with structural data, links and advertisements, commonly called *Text Extraction*. This work is particularly challenging because of the difficulty in determining which part of a Web page is meaningful and which part is not. Despite the importance of this topic, little research has been done so far and current methods make too many assumptions. In this paper, we propose an effective heuristic for extracting meaningful content from Web pages called the *Text-to-Tag Ratio* (TTR).

Our technique is based on the observation that all Web pages have some structure and that this inherent structure varies greatly. The essence of our technique can be seen by viewing the HTML-source of any Web page. We observe that most Web pages contain a title banner on the top with a list of hyperlinks on the left or right side of the page with advertisements interspersed. Most usually the meaningful content of the page is located in the middle. Of course, this layout is not standard among all Web pages, and this fact is the crux of our approach.

After reviewing the state of the art, this paper will introduce TTR and give examples of its use. Next, we smooth the TTR histogram and then cluster the results into content and non-content sections. Finally, we test the results of our approach by comparing the computed content clusters to human-generated ground truth. Our main empirical objective is to maximize recall because we view the extraction of errant content to be less detrimental than the exclusion of actual content. Space savings will also be shown as a result of the text extraction.

II. Related Work

Internet text extraction is an important problem and yet little research has been done in this area.

A naive solution is to simply remove all HTML tags. However this approach allows structural and non-content related text such as site-links, advertisements, copyright information, etc. to remain.

In [6], a method is proposed to discover informative contents from pages of a Web site. Their use of entropy is similar to our use of the text-to-tag ratio, but their approach is limited by the following assumptions as documented in [7]: (1) the system knows *a priori* how a particular Web page is formatted; and (2) the system assumes that all content appears within the `table` tag. As we shall see, these two assumptions are difficult to make especially given the trend that Web page developers are moving away from `table` tags and towards `div` tags with the use of CSS as documented in [8]. Our approach does not rely on the use of any particular HTML tag(s).

In [7], Web page cleaning is defined as a frequent template detection problem. They propose that by counting frequent item sets they can discover web templates and extract data content from a Web page by applying a template, and then by finding the particular *pagelet* that the page-content can be extracted from. This approach assumes that all Web pages can be categorized to fit a particular template, and furthermore, that the *pagelet* that was selected from the template contains all of the content from the page. Our approach makes no assumptions on the particular style or structure of a web page. Our only assumption is that the Web page in question has *some* structure.

In [8], the authors describe a method based on the analysis of both the structure and the content of the Web pages in a given Web site. Again some assumptions are made with this approach: (1) the system must process one contiguous Web site at a time in order to describe the common structure of a set of Web pages; and (2) the system assumes that all Web pages in a Web site are similar. However, while this is often the case it is not a mandatory condition. For instance, the various departments and offices at Kansas State University[1] each have their own style and structure. Furthermore, student Web pages share little in common with the university or department structure although they technically remain part of the university's domain. Our approach does not assume a particular structure is shared among Web pages of the same Web site.

The common problem with the related work is that too many assumptions are made. This is true especially for the structures of the Web pages in question.

III. The Text to Tag Ratio

We, thus, introduce, the *Text-to-Tag Ratio* (TTR) as the basis by which we analyze a Web page in preparation for text extraction. It, essentially, is the ratio of the count of non-HTML-tag characters to the count of HTML-tags per line. In the likely event that the count of HTML-tags on a particular line is 0 the ratio is set to the length of the line. The TTR algorithm is described in Alg. 1.

Algorithm 1: *Text-to-Tag Ratio* pseudocode

```
input
  $h \leftarrow$ HTML source code
begin
  Remove all script and remark tags and empty lines
  for each line $k$ to $numLines(h)$ do
    $x \leftarrow$ number of non-tag ASCII characters in $h[k]$
    $y \leftarrow$ number of tags in $h[k]$
    if $y = 0$ then
      $TTRArray[i] \leftarrow x$
    else
      $TTRArray[i] \leftarrow x / y$
    end if
  end for
  return $TTRArray$
end
```

Before TTRs are computed, `script` and `remark` tags are removed from the HTML document because this information would be treated as non-tag text by the algorithm and thus skew the results. Empty lines are also removed because their inclusion would potentially skew the performance of the clustering algorithms that are described in Section IV.

[1] The Kansas State University Web site is available at http://k-state.edu

As an example, consider a news article from The Hutchinson News[2] that appeared on Wednesday, March 19, 2008. This Web page is similar to many pages on the Web. The title banner, hyperlinks and advertisements take up most of the space on the Web page while the content of the page is confined to a relatively small space in the middle. At the bottom of the page more advertisements and images are displayed along with links to copyright and other administrative information.

When this page's source is analyzed with the TTR algorithm, our initial estimation becomes evident because a large cluster of high TTR lines is seen between lines 225 and 262 as shown in Fig 1. This area of high TTRs denotes the content section of the Web page.

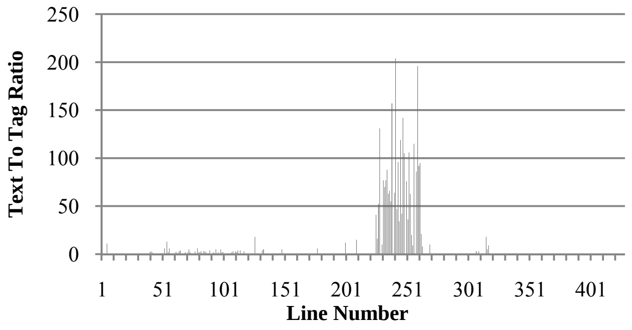

Figure 1: Text to Tag Ratio for a Web page from The Hutchinson News

IV. Computing Clusters

Now that the *Text-to-Tag Ratio* (TTR) has been computed for each line we turn our attention to the task of clustering the resulting *TTRArray* in order to determine the content lines of the Web page. For this purpose the TTR Heuristic is considered.

TTR Heuristic:

For each k in *TTRArray*, the higher the TTR is for an element k relative to the mean TTR of the entire array the more likely that k represents a line of content-text within the HTML-page.

In this section we describe four clustering techniques based on the above heuristic, namely, K-Means, Expectation Maximization (EM), Farthest First, and two thresholding techniques that were developed specifically for this problem.

The standard deviation is used as the clustering threshold because it best represents the variance of the array while the mean is more likely to be skewed by large outliers and the median is more likely to pick a point that is too low because of the preponderance of low TTR scores among the lines of most Web pages.

Pre-processing

Before any clustering techniques are applied to the array a smoothing pass is made on the array. This is done because without smoothing, many important content-lines might be lost including the article title, by-line, short or one sentence

[2] The Hutchinson News is available online at http://hutchnews.com. The specific article is not permanently linked.

paragraphs, etc. that would otherwise fall below the clustering threshold.

As a pathological example, consider the American Declaration of Independence[3]. As an HTML document the TTR would heavily favour the preamble and proclamation sections but would lose some of the abuses of King George III because the abuses are listed in a short, concise manner, and relative to the rest of the document their TTRs lie below the threshold as shown in Fig. 2.

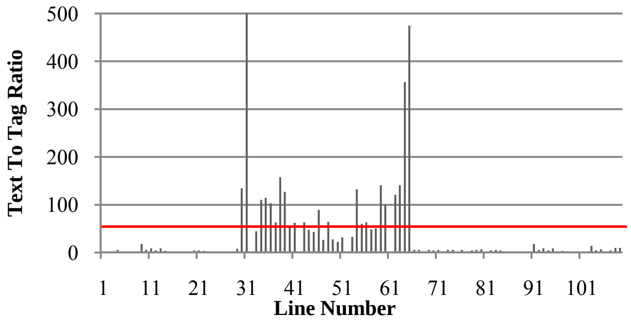

Figure 2: Text-to-Tag Ratio for an American Declaration of Independence Web page[3]. Horizontal line denotes the threshold at 1 standard deviation (64.49 TTR).

To resolve this problem we smooth the TTR array by computing the mean TTR (e) for a given radius (r) for each element (k) for all lines (n) as shown in Eq. 1.

$$e_k = \frac{\sum_{i=k-r}^{k+r} TTRArray_i}{2r+1} \qquad (1)$$

For all experiments included in this paper we use a radius of size 2 excluding k.

The resulting array, shown in Fig. 3, is better suited for clustering because of the increased cohesiveness within sections and strict differences between sections. Furthermore, the resulting array has a lower standard deviation of 40.55 TTR because outlying peaks and valleys are normalized.

Also, outliers, such as advertisements, that may occupy one line among many low-TTR lines are smoothed out and not counted.

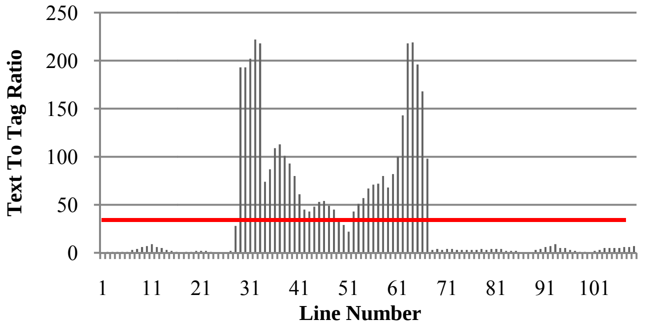

Figure 3: Smoothed Text to Tag Ratio for an American Declaration of Independence Web page[3]. Horizontal line denotes the threshold at 1 standard deviation (40.55 TTR).

[3] The copy of the American Declaration of Independence used in this paper is available online at http://www.ushistory.org/declaration/document/index.htm

K-Means, Expectation Maximization, Farthest First

To cluster the smoothed array into content and non-content clusters we consider three well documented clustering algorithms: K-Means, Expectation Maximization (EM), and Farthest-First. For the purposes of our study we will be using the implementations as provided as part of *WEKA*.

K-Means is a well known iterative distance-based clustering algorithm; it also is one of the oldest, simplest and most widely used clustering algorithms [9]. EM is a statistical model that makes use of the finite Gaussian mixture. Detailed descriptions and numerous references regarding this topic can be found in [10]. Farthest-First is a clustering algorithm that combines hierarchical clustering and distance-based clustering. In particular, it uses the basic concept of agglomerative hierarchical clustering in combination with a distance measurement criterion that is similar to the one used by K-Means [11].

Because of the diversity of Web pages, the number of clusters to compute could not remain constant. With this in mind, we executed each clustering algorithm 3 times for each Web page – one execution for 1 through 3 clusters[4]. For each execution we recorded the number of clusters that had a mean above the standard deviation of the entire array. The lines that are included in those clusters are deemed to be content-lines.

For example, consider the news article from Section III. After smoothing the standard deviation is 20.35. For a single cluster the K-Means algorithm returns in zero hits as shown in Table 1. For two clusters the K-Means algorithm returns a single cluster because the mean of that cluster is 53.40, which is greater than the standard deviation threshold of the array. Finally, for three clusters the K-Means algorithm again returns a single cluster. Because there is a tie between choosing 2 and 3 clusters an arbitrary decision is made. The default decision goes to the lower cluster number.

Cluster	1 cluster	2 clusters	3 clusters
1	6.85	0.56	10.12
2	-	**53.40**	**70.42**
3	-	-	0.59

Table 1: Means of clusters for K-Means on an article from Hutchinson News. Bold denotes means above the standard deviation. 2 clusters are selected.

After the cluster(s) are selected the corresponding lines are selected from the HTML document. Each line is then stripped of all remaining HTML tags – usually `paragraph` and `anchor` tags. Finally, the cleaned lines are combined and output to a file for storage.

Threshold Clustering

Another approach we use to cluster a smoothed TTR array into content and non-content sections is to apply a threshold of one standard deviation to a smoothed TTR array. Array elements with a TTR that lies equal to or above the standard

[4] We chose a maximum of 3 clusters arbitrarily. Specific clustering techniques and the results therein are outside the scope of this paper.

deviation shall be deemed content-lines, and those which are less than the standard deviation are considered non-content lines.

Again, consider the example from Section III. The article's unsmoothed TTR array is shown in Fig. 1. After smoothing, a threshold is applied at 1 standard deviation above the base line. This results in an easy distinction between content and non-content lines as shown in Fig. 4.

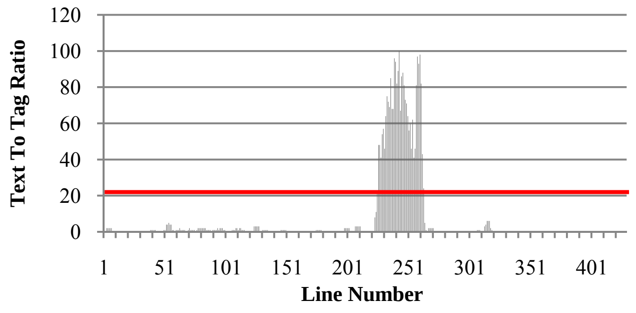

Figure 4: Smoothed TTR array of Hutchinson News article with a threshold at 1 standard deviation (20.30) above the base line.

Prediction Clustering

Our final clustering approach is very similar to the threshold clustering discussed in the previous subsection, in that, prediction clustering operates on the assumption that the TTR of the content lines of Web pages appear in stark contrast to the TTR of non-content lines. By operating under this observation we iterate through a non-smoothed TTR array and attempt to "predict" whether or not the movement from one line to another constitutes a "jump" from a non-content section to a content section of the Web page and vice versa.

Specifically, our prediction algorithm looks for jumps of 1 standard deviation over the moving average of the previous 3 TTRs as an indication to the detection of a content section. Inversely, to exit a content section, the prediction clustering algorithm looks for a reduction in the mean of the next 3 TTRs of at least 1 standard deviation from the current, in-content, TTR.

We believe that this approach is potentially the most effective at maximizing recall because the short content lines that lie beneath the threshold – as described in Section IV – have a smaller chance of being ignored.

V. Experimental Results

In this section we demonstrate the results of applying the TTR technique from Section IV and the various clustering techniques from V.

To test the accuracy of the *Text-to-Tag Ratio* techniques documented in the above sections, 176 complete Web pages were downloaded. These pages were selected by searching for the keyword "the" from Yahoo's search engine and harvesting the results. Note that while a total of 200 Web pages were retrieved, 34 had to be discarded because they were either incompatible files, such as PDFs, etc. (31 occurrences), or they were blank (3 occurrences). No linked or referenced files, including style sheets, images, etc., were included in the test corpus.

The goal of our experiments was to determine the content data of the Web pages and filter out all extraneous advertisements, site links, etc. We determined the actual content of each Web page by opening each downloaded file in a browser and manually selecting the content text. The text was copied into a new file and is used for comparison evaluation later.

Evaluation Metrics

To test the TTR and clustering techniques, methods were devised to compare the algorithm-results to the manually determined ground truth.

The first test metric is the Longest Common Subsequence (LCS) [12]. The LCS determines how much of the ground truth was obtained by our algorithm by comparing the longest common subsequence of both files. Before the comparisons are made all line breaks, special HTML characters, and extra white spaces are removed from both files because a single errant character can have disastrous effects on the LCS score. The results of this metric are recorded as the percent of the algorithm output that matches the longest common subsequence of the manual file to the total length of the manual file. For the purposes of these experiments we consider LCS to be recall, and the mean LCS for all pages is reported in these results.

The second and final test metric is the edit distance ratio (EDR). The edit distance measures how many key strokes are necessary to transform the algorithm output into the manual file [13]. Like in the LCS metric, all line breaks, whitespace, etc. is removed. For our purposes, EDR is the ratio of edits needed to transform the algorithm output (o) into the manual file (m) to the total length of the longest file as shown in Eq. 2.

$$EDR = \frac{1 - \text{edtDist}(o, m)}{\max\big(\text{len}(o), \text{len}(m)\big)} \tag{2}$$

Because of this ratio we actually wish to maximize EDR, and for the purposes of these experiments we consider EDR to be precision. The mean EDR for all pages is reported in these results.

Results and Interpretation

The chart in Table 2 shows the LCS results for each clustering method. The threshold clustering method achieved the best results with a mean of 94.19% and a competitive median.

	Threshold	EM	K-Means	Far. First	Prediction
Mean	**94.19%**	92.62%	92.47%	85.88%	81.14%
Median	98.65%	**99.34%**	98.68%	94.18%	94.42%
Std Dev.	**14.03%**	17.60%	16.57%	21.32%	24.85%
Matches	34	**43**	35	25	22

Table 2: Mean and median LCSs for clustering methods, where higher is better for LCS and matches, and lower standard deviation is better. Winners are in bold.

EM achieved the highest number of perfect matches with ground truth, on 43 of 176 documents.

Table 3 shows the EDR results for each clustering method. The farthest first clustering method achieved the best results with a median and mean of 77.03% and 62.53% respectively. There were no perfect EDR matches.

	Threshold	EM	K-Means	Far. First	Prediction
Mean	56.21%	48.77%	57.44%	**62.53%**	52.40%
Median	61.63%	48.98%	61.17%	**77.03%**	55.30%
Std Dev.	31.89%	30.66%	32.96%	33.75%	**30.01%**

Table 3: Mean and median EDRs for clustering methods, where higher is better for EDR, and lower standard deviation is better. Winners are in bold.

As discussed in Section V, the threshold clustering technique from Tables 2 and 3 have a threshold that is set to 1 standard deviation above the base line. Fig. 5 shows that as the threshold increases from a standard deviation coefficient of 0.0 to 2.0 the LCS (recall) decreases and the EDR (precision) increases.

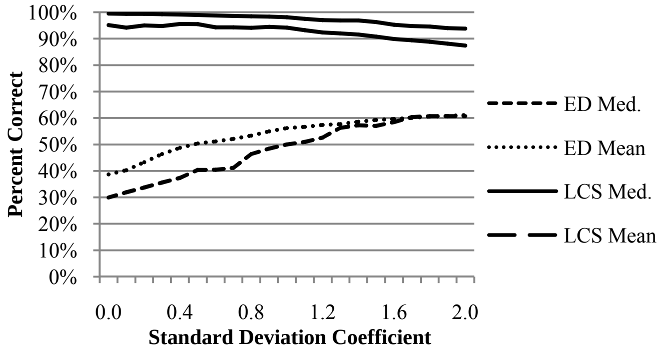

Figure 5: LCS and EDR metrics for increasing coefficients on the standard deviation for the Threshold of the TTR array.

We are also able to show a mean space savings of 95% when the extracted text is compared to the original HTML. Although dictionary compression schemes, such as GZip, significantly compress HTML documents, we are still able to show an 88% space savings when the compressed HTML is compared to the compressed text. The mean results are shown in Table 4.

	HTML	Extracted Text	GZip HTML	GZip Text
File Size (Kb)	9,630.34	497.70	2,234.77	275.53

Table 4: Mean file sizes for 176 original HTML files compared to the mean file sizes for the extracted text.

VI. Conclusions

In this paper we discussed the problem of extracting the content text from diverse Web pages without the use of specific tags or other specific HTML cues. We showed that current approaches are too specific and do not account for the variety of Web presentation styles, especially with the increased use of CSS. We proposed a new method of text extraction using the *Text-to-Tag Ratio* of each line of the HTML document, and then using various clustering techniques to identify the content sections of the document. We then showed empirical results that indicate EM and threshold clustering techniques are best able to correctly identify content sections of an HTML document. Finally, we showed that by extracting the content text from the original HTML document we can clean miscellaneous hyperlinks, advertisements, etc. with high recall for all levels of precision and a large space savings.

We believe that further refinement of the prediction clustering algorithm will yield better results. In the future, we intend to extend our prediction clustering algorithm using density-based approaches, in order to more accurately predict changes between content and non-content sections of HTML documents. Furthermore, we will explore this approach in conjunction with other smoothing techniques and investigate the application of this text extraction technique to augment existing text summary, sentiment analysis, etc.

Acknowledgements

We would like to thank Dr. Dan Andresen and Dr. Doina Caragea for their insight and useful comments, and Daniel Jones, John Drouhard, Imran Hameed and Jack Hart for their assistance with this project.

References

[1] W. Lu, L. Chien, and H. Lee, "Anchor text mining for translation of Web queries: A transitive translation approach," *ACM Transactions on Information Systems (TOIS)*, vol. 22, no. 2, pp. 242-269, April 2004.

[2] S. Soderland, "Learning to Extract Text-based Information from the World Wide Web," in *Proc. Of KDD 1997*, Newport Beach, California, USA, 1997.

[3] R. J. Mooney, and R. Bunescu, "Mining Knowledge from Text Using Information Extraction," in *Proc. of SIGKDD 2005*, Chicago, Illinois, USA. Aug. 2005.

[4] M. Y. Ivory, R. Megraw, "Evolution of web site design patterns." *ACM Transactions on Information Systems (TOIS)*, vol. 23, no. 4, pp. 463-497, Oct. 2005.

[5] D. C. Rajapaske, and Jarzabek, "A Need-Oriented Assessment of Technological Trends in Web Engineering," in *Proc. of ICWE 2005,* Sydney, Australia, July, 2005.

[6] S. H. Lin and J. M. Ho. "Discovering informative content blocks from Web documents," in *Proc. of SIGKDD 2002*, Edmonton, Canada, 2002.

[7] Xiaoli Li, Bing Liu. "Learning to classify text using positive and unlabeled data." in *Proc. of IJCAI 2003*, Acapulco, Mexico, 2003.

[8] B. Krüpl, M. Herzog, and W. Gatterbauer, "Using visual cues for extraction of tabular data from arbitrary HTML documents," in *Proc. of WWW 2005*, Chiba, Japan, 2005.

[9] J. B. MacQueen, "Some Methods for classification and Analysis of Multivariate Observations," in *Proc. of 5th Berkeley Symposium on Mathematical Statistics and Probability*, Berkeley, CA, 1967.

[10] A. Dempster, N. Laird, and D. Rubin. "Maximum likelihood from incomplete data via the EM algorithm". *Journal of the Royal Statistical Society*, Series B, vol. 39, No. 1, pp. 1-38, 1977.

[11] S.D.Hochbaum, B.D. Shmoys, "A Best Possible Heuristic for the KCenter Problem," *Mathematics of Operational Research*, vol. 10, no. 2, May, pp. 180-184, 1985.

[12] D. S. Hirschberg, "Algorithms for the Longest Common Subsequence Problem," *Journal of ACM*, vol. 24, no. 4, pp. 664-675. 1977.

[13] V. Levenshtein, "Binary Codes Capable of Correcting Deletions, Insertions, and Reversals," *Soviet Physics-Doklady* 10, vol. 10, pp. 707-710, 1966.